NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

THE EXTENSION OF UNIT ALLOCATION AND COUNTERMOBILITY PLANNING ALGORITHMS IN THE AIRLAND RESEARCH MODEL

by

Joseph R. McLaughlin

March 1986

Thesis Advisor: Samuel H. Parry

REPORT DOCUMENTATION PAGE

REPORT SECURITY CLASSIFICATION CLASSIFIED	1b. RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY DECLASSIFICATION / DOWNGRADING SCHEDULE	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
ERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
NAME OF PERFORMING ORGANIZATION val Postgraduate School — 6b OFFICE SYMBOL (If applicable) 74	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
ADDRESS (City, State, and ZIP Code) nterey, California 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
NAME OF FUNDING / SPONSORING ORGANIZATION — 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

ITLE (Include Security Classification)
E EXTENSION OF UNIT ALLOCATION AND COUNTERMOBILITY PLANNING ALGORITHMS IN THE AIRLAND SEARCH MODEL

ERSONAL AUTHOR(S)
Laughlin, Joseph R.

TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
ster's Thesis	FROM ______ TO ______	1986	106

UPPLEMENTARY NOTATION

COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	ALARM, AirLand Research Model, Network Combat Simulations, Maneuver Unit Allocation, Engineer Asset Allocation, Ground Combat Planning Model

BSTRACT (Continue on reverse if necessary and identify by block number)
is thesis extends the development of the AirLand Research Model (ALARM), an on-going search effort at the Naval Postgraduate School (NPS), in the areas of maneuver unit location and countermobility planning at the battalion level. The feasibility and sirability of multiple algorithms to determine enemy avenues of approaches into a ttalion sector and to select the optimal position along the avenue for unit placement demonstrated. The concept of analyzing terrain on the basis of flow rate, the ility to through-put a deployed attacker, is investigated. An algorithm linking unit acement and counter-mobility operations is developed. Additionally, shortcomings ted in ALARM by previous research are resolved. These developments have been coded to SIMSCRIPT and integrated into the existing model on the VAX 11/780 at NPS.

ISTRIBUTION / AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified	
NAME OF RESPONSIBLE INDIVIDUAL nuel H. Parry	22b TELEPHONE (Include Area Code) 408-646-2779	22c OFFICE SYMBOL 55Py

ORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

The Extension of Unit Allocation and Countermobility Planning Algorithms in the AirLand Research Model

by

Joseph R. McLaughlin
Captain, United States Army
B.S., United States Air Force Academy, 1976

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY
(Command, Control and Communications)

from the

NAVAL POSTGRADUATE SCHOOL
March 1986

ABSTRACT

This thesis extends the development of the AirLand Research Model (ALARM), an on-going research effort at the Naval Postgraduate School (NPS), in the areas of maneuver unit allocation and countermobility planning at the battalion level. The feasibility and desirability of multiple algorithms to determine enemy avenues of approaches into a battalion sector and to select the optimal position along the avenue for unit placement is demonstrated. The concept of analyzing terrain on the basis of flow rate, the ability to through-put a deployed attacker, is investigated. An algorithm linking unit placement and counter-mobility operations is developed. Additionally, shortcomings noted in ALARM by previous research are resolved. These developments have been coded into SIMSCRIPT and integrated into the existing model on the VAX 11/780 at NPS.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

A. THE AIRLAND RESEARCH MODEL

This thesis represents a continuation of the development of the AirLand Research Model (ALARM) at the Naval Postgraduate School (NPS). ALARM will be a corps level, force-on-force simulation that will allow development and testing of AirLand Battle doctrine. This doctrine calls for defeating an enemy attack by the combined efforts of destroying his leading combatants, preventing the effective deployment of second echelon forces, and interdicting forces attempting to move up to the battle area. Operations at extended ranges and close Air Force-Army cooperation are foundations of this doctrine.

Towards this goal, ALARM seeks to develop several methodologies previously not incorporated in production military models. ALARM will be designed to operate in a systemic mode, without a man-in-the-loop. This goal has led to the formulation of a two-tiered architecture. ALARM will have a Planning Model, and a separate, distinct Execution Model. The development of high-resolution execution models is well advanced and many methodologies exist for implementation into ALARM. The focus of current research at NPS has been on the Planning Model. The Planning Model will contain rule-based systems that represent the planning function from corps down to battalion. Output of the planning process will include mission, sector, and task organization.

A cornerstone of ALARM's development has been the use of network structures to represent terrain and planning processes. This methodology promises great storage and computational savings over techniques such as hex, digital, or functional terrain. Network structures will facilitate aggregation of data and processes as the level of resolution

shifts from battalion to corps. This thesis will deal strictly with the terrain network. The terrain network is composed of arcs and nodes that represent routes of movement and terrain features, respectively. An example of the network is displayed in Figure 1.1 . This area represents a 25x75 Km area west of Fulda, in the Federal Republic of Germany. See Craig [Ref. 1: pp. 37-47] for a complete description of the terrain network. See Appendix A for description of terrain network arcs and node characteristics.

Another methodology being explored by ALARM research is the Generalized Value System (GVS). AirLand Doctrine requires engagement of enemy forces at extended ranges from the front. Currently, no satisfactory method exists to establish the value, and hence engagement priority, of distant targets. The GVS attempts to solve these problems by determining a unit's value based upon the amount of time before it can affect the battle.

B. PURPOSE

The purpose of this thesis is to extend the development of the Planning Model in the area of ground force mission generation and disposition. This effort is an outgrowth of previous students' theses at NPS. Boyd [Ref. 2] proposed a manuever unit positioning scheme at the battalion level and Kazimer [Ref. 3] developed a system for allocating engineer counter-mobility resources. Craig [Ref. 1] implemented these theories on a model he developed based upon Krupenevich's work [Ref. 4] on network structures for combat models.

The construction of the terrain network and resultant ability to test allocation algorithms has greatly facilitated the ALARM research effort. The validity of previous assumptions and algorithms is now much more easily analyzed with the aid of a working model. Student research has been

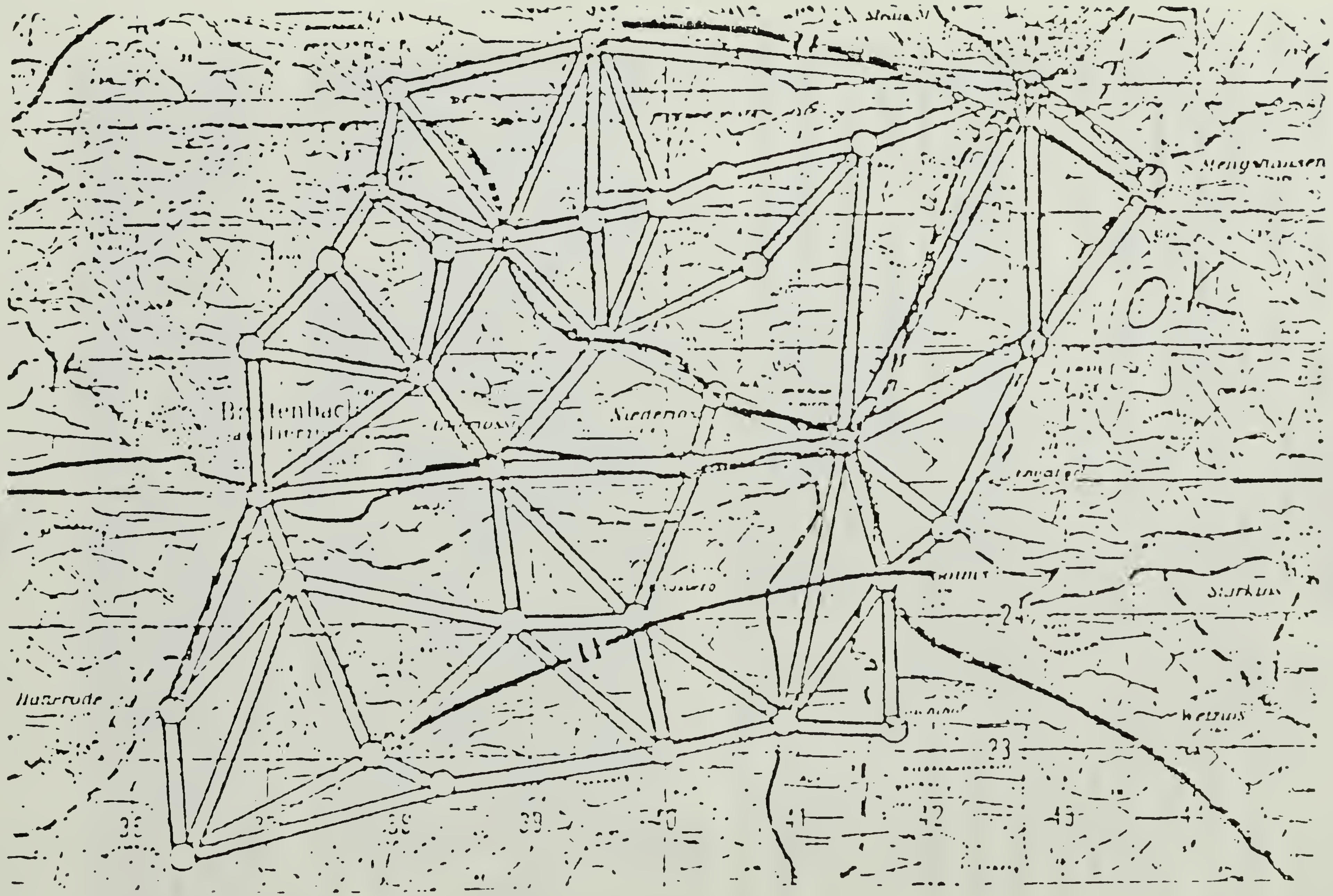


Figure 1.1   Network Representation.

further assisted by the work of Rolands and Associates, Inc., Monterey, CA in converting and improving Craig's model written in TURBO PASCAL for an IBM PC to Simscript routines on the VAX 11/780 in the Wargaming Analysis and Research (WAR) Lab at NPS.

An analysis of previous research efforts reveals a need for algorithm refinement in several areas. These shortcomings fall generally in the areas of avenue of approach generation and arc selection. It should be noted that efforts to date, as well as this thesis, focus on the defense scenario.

Efforts for determining the enemy's avenue of approach through a battalion sector have centered on finding the minimum time path from the Forward Line of Troops (FLOT) to the sector rear boundary. This approach presupposes a knowledge of the enemy's objective. It is assumed the enemy's objective lies distant to the rear. In the general case, this may prevail. However, depending on the size of the battalion sector and other factors, alternative objectives are certainly possible. For example, the enemy may have as his objective a location in the current sector, or the destruction of forces within the sector. An estimation of the enemy's mission and objective is vital in selecting the most appropriate avenue of approach optimization routine. Contributing to this analysis will be an important task of the intelligence network of the Planning Model, an area of future research. In the meantime, the development of realistic alternative avenue selection algorithms will be a goal of this thesis.

The concept of a minimum time path appears valid as long as one is very precise about what time is being evaluated. Is time attained by dividing a road's length by an average speed or should off-road trafficability or road type come in to play? Craig [Ref. 1: p. 56] attempts to capture the

effects of off-road speed in his equation for arc traversal time. This process may unduly favor a narrow improved road through otherwise impassable terrain. Consideration must be given to the enemy's preferences for, all other things being equal, a route offering room to deploy and manuever on contact. Manzo [Ref. 5: p. 46] introduces the concept of a cross-country flow rate equation:

$$\text{Flow Rate} = (Va * Wc) / (Wd * Ld) \qquad \text{(eqn 1.1)}$$

where:

Flow Rate = (Bn/Hr).

Va = Average speed permitted along the arc (Km/Hr).

Wc = Width of arc for vehicle deployment (Km).

Wd = Doctrinal width of battalion (Km/Bn).

LD = Doctrinal length of battalion (Km).

This flow rate, when divided into the number of battalions in the enemy's assault, provides a time for route travel that is more descriptive than that previously considered. Equation 1.1 considers an arc's terrain and its impact on combat. This thesis develops an equation for avenue minimum travel time that incorporates the effects of flow rate.

As Craig points out [Ref. 1: p. 63], the time value of an arc is not just the time required to traverse it. The occupation of an arc by a defender and the resultant battle creates an additional amount of time that must be modelled. Craig solves for battle duration with a Lanchester Linear Law formulation:

$$T = (1/(V1-V2)) * \ln ((V1-(1-DBP) * V2) / (ABP * V1)) \qquad \text{(eqn 1.2)}$$

where:

V1 = (Attacker Standard Unit of Armament (SUA)/Defender SUA) * Attacker Rate of Fire

V2 = (Defender SUA/Attacker SUA) * Defender Rate of Fire

ABP = Attacker Breakpoint (expressed as a fraction of initial strength)

DBP = Defender Breakpoint (expressed as a fraction of initial strength)

In this case, the attacker is given credit for all of his unit's firepower at the start of the battle. He is allowed, in effect, instant deployment and the ability to bring all of his forces to bear. Terrain limitations cannot be represented. An alternate solution would be an equation for time that captures the ability of the defender to select the terrain optimal for his defensive plan. This thesis develops an equation of this type.

In addition to avenue of approach generation, the arc selection process is an area requiring refinement. Previous efforts optimized a single arc attribute such as line of sight or acquisition range. The best avenue of approach was then scanned for the arc with the highest value of that attribute. An obvious problem arises if a sector has large open areas in its rear.

Arc selection, in effect unit positioning, is a much more complicated process than can be effectively described by a single physical arc attribute. Like avenue of approach generation, arc selection algorithms must at a minimum consider mission, friendly forces, and the enemy. Higher echelons may direct certain objectives to be defended, which simplifies the arc selection process. A defending force composed of Light Infantry, with relatively few long range weapons, would be at a distinct disadvantage if it selected an arc that optimized line of sight against an armor-heavy attacker. Clearly, arc selection algorithms must be sensitive to the specific tactical situation.

Additionally, there is no reason why the planning model should not take advantage of a capability it has that the human planner lacks: the ability to quickly simulate arc

combat and receive immediate feedback on one arc's combat potential vice another. The situation could arise where casualty ratios or protection of the defending force may be the criteria by which arcs must be evaluated.

The arc selection process is a dynamic one affected by a number of variables. In order to produce effective plans for unit placement, the planning model must consider these variables. The development of algorithms to accomplish this will also be a goal of this thesis.

In addition to refining avenue generation and arc selection algorithms, this thesis addresses problems in two other areas. Craig's algorithms did not reflect the ability of a unit to exert its influence over more than one arc. Without this ability, three companies might be placed on different arcs emanating from a common node. In a real situation, one company would probably be placed on the node to cover all three arcs. A solution to this problem will be developed.

The last major area to be addressed concerns unit and counter-mobility obstacle placement. Efforts to date have essentially treated these as two distinct processes. Units were placed on the network with one algorithm and engineer obstacles were placed by another. There was no guarantee of coverage of obstacles by fire. While this unfortunately models occasional practice in the field, it is not the optimal method for a planning model. Craig recognized this and attempted to ensure obstacles were placed in front of the units and thereby hopefully within their line of sight [Ref. 1: p. 24].

As discussed previously, the avenue generation and arc selection processes must consider the nature of the terrain. Likewise, obstacle placement must consider the ground tactical plan. An obstacle not covered by fire is merely a ditch-filling or river-crossing drill for enemy engineers. Engineer resources are too scarce to allocate them in a

manner that does not ensure their integration into the ground tactical plan. This thesis develops an algorithm that integrates these two processes.

C. METHODOLOGY

In attempting to achieve the goals of this thesis, the defense planning process will be examined in Chapter II to ensure that the most important areas are considered. The terrain network will then be analyzed in Chapter III to determine how it can contribute to an evaluation of defense fundamentals. Model Components will be developed in Chapter IV. Algorithms have been coded in Simscript and tested on the working model in the NPS WAR LAB. Chapter V will discuss the results of this testing. Chapter VI will contain an analysis of the Humboldt equation used to compute attrition in the revised arc selection process. Finally, Chapter VII will summarize the results of the research.

II. THE DEFENSE PLANNING PROCESS

A. KNOWLEDGE REQUIRED FOR PLANNING

In the ground environment, tactical planning currently represents an integration of doctrine and what is known as METT-T (Mission, Enemy, Troops, Terrain, and Time). Doctrine prescribes the general principles of 'how to fight' one's forces. It is a broad analysis of the current state of the art of warfare and attempts to produce guidelines for the conduct of operations that will improve the likelihood of victory. METT-T represents those local conditions that determine the specific manner in which doctrine will be applied. The end result of synthesizing these two separate bodies of knowledge is tactics.

This review of the foundations of tactical planning is beneficial because it represents those activities that ALARM must at least approximate if it hopes to generate effective and credible plans. Its decision rules must be grounded in current doctrine, otherwise resultant plans may be incomprehensible to the commander who is asked to implement them.

METT-T must be considered in the tactical planning if one hopes to differentiate among the endless tactical alternatives available. Understanding the mission will go a long way towards determining what kind of path and arc to optimize. As mentioned earlier, the mission may be to defend an objective such a road junction, in which case unit placement is essentially determined by higher echelon. The mission may be to conduct a delay from successive phase lines for a specified time. Planning algorithms must be able to interpret mission requirements, because they will have a great impact on what kinds of plans will be feasible.

Likewise, the enemy must be considered. What does his doctrine prescribe? What are the size and composition of his forces? How will he reinforce? These issues determine, in the present analysis, what kind of avenues and arcs to optimize in the planning process.

Troop and resource availability is critical to the planning process. Certain plans will be infeasible for certain units. The availability of engineers and fire support, to mention just two forms of combat support, might transform a poor option into an attractive one.

The importance of terrain to planning cannot be overemphasized. Virtually the first thing a commander does after receiving his mission is to study his map. This, and visual reconnaisance, allow the commander to mentally image the battle and assists in formulating his plan. The planning model must also be able to analyze the terrain from a tactical perspective. It must know certain essentials for ground combat planning such as trafficability, cover, concealment, and fields of fire. If terrain is not considered, or is improperly interpreted, the resultant plan will have substantially reduced effectiveness.

Time is another element that impacts on the tactical plan. Here it represents the amount of time available to prepare for the mission. If a defender had little time to get to his sector and prepare a defense, then tactics which require a significant amount of time such as a strongpoint or a deliberate defense would probably not be feasible.

Lastly, weather should be considered in formulating the tactical plan. Visibility and trafficability restrictions caused by weather can require significant tactical changes.

The tactical planning process requires input from numerous sources. It must be able to work within the framework of service doctrine to have an idea of 'how to fight' and generate plans that are credible in execution. It must

also receive and analyze information on the current situation. The numerous required inputs and alternative outputs indicate a need for the planning process to have flexible and varied algorithms.

B. DETERMINING THE AVENUE OF APPROACH

Having received his mission and sector, the defender's first task is to determine where the enemy is likely to go. Hopefully, the enemy's objective will coincide with the unit's object of defense. When they do, the defender can most effectively conduct his defense. If the defending unit's analysis indicates they do not coincide, then the higher planning echelon must be informed for possible mission revision. For example, a commander given the mission to defend a bridgehead may determine that numerous fording sites exist in his sector, unknown to higher echelon. This must be reported so adjustments can be made to ensure the water obstacle is not easily crossed. The enemy's objective may be within, or to the rear of, the sector. Intelligence from different echelons will help determine this.

Once the objective is estimated, the defender must estimate what route the enemy will use to attain his objective. Enemy composition and doctrine must be considered in this estimation. If the objective is determined to be a bridge, if enemy doctrine includes airmobile assaults, if and local enemy forces have air assault assets, then the defender may consider the primary avenue of approach to be an air corridor and plan a detailed air defense. Another example would occur where the objective was a hill top, enemy forces consisted mainly of Light Infantry and enemy doctrine called for extended foot marches at night in the offense. In this case, the avenue of approach might be the best covered and concealed route to the objective.

C. GENERATION OF SECTOR AVENUES OF APPROACH

Having determined the enemy's probable objective as well as his path to get there, the network can be evaluated for those arc and node attributes which would be useful in developing mathematical treatments to determine the enemy's expected route of travel. If the attributes are available, algorithms can be developed to optimize virtually any kind of route desired. In the previous Light Infantry example, an algorithm optimizing covered and concealed routes might perform operations on the arc attribute that describes arc type. An arc type coded forest might be weighted with a low number while an arc type coded open country would be weighted with a high number. This would allow a path solving technique such as Dysktra, which is currently employed in ALARM, to favor avenues with the low, forest arc-type numbers. With a variety of algorithms developed, the Planning Model would only need to decide which one was most appropriate for the situation. Several of the more applicable algorithms will be developed in Chapter III.

D. POSITION SELECTION

The next step in the planning process for the defender is to decide where to fight along the anticipated enemy path. How the defender can fight will be a key determinant. These tactical decisions are affected by the synthesis of doctrine and METT-T, as discussed previously. As with avenues of approach, algorithms can be developed from applicable arc and node attributes. They may range from an evaluation of a single attribute to more detailed functions. Since relatively few arcs are involved along a given avenue of approach, combat may be simulated and the results used in the solution algorithms. Several algorithms will be developed in Chapter III.

III. OPTIMIZATION METHODOLOGY

A. POSITION OPTIMIZATION SCHEMES

A crucial process in the placement of units on the terrain network is arc evaluation. One may estimate the enemy's avenue of approach correctly every time, but if an infeasible battle position is selected, the defender will fail in his mission. Previous efforts have selected the best arc on the basis of scanning a single attribute such as maximum acquisition range on every arc in the avenue of approach. This section will develop a more detailed evaluation method. For the following discussion, it is assumed the best avenue has already been determined. The next section will consider algorithms for avenue of approach generation.

As discussed in Chapter II, the decision of where and how to fight is much more complicated than can be adequately described by a single physical attribute. This section will develop two methods of arc selection which will hopefully better model actual planning considerations.

1. General Considerations for a Defensive Position

In the general case, the defender wants to pick a position from which he stands the best chance of defeating the enemy. In the present context, this will be considered achieved if the defender attrits the attacker to his breakpoint. The breakpoint is that fraction of the attacker's initial strength that upon being reached, the attacker will no longer conduct offensive operations. In the absence of breaking the attacker, the defender would want to delay the enemy as much as possible. This can be achieved by inflicting the maximum casualties possible on the attacker and choosing a battle site that causes a lengthy battle to develop. It may also be achieved by trading terrain for

time. As the offensive planning algorithms are developed, counter-attacks will be available as a course of action for the defender.

The need then arises to simulate the combat between attacker and defender on every arc along the path. Casualties from the combat will determine if breakpoint is reached and permit the calculation of added delay time to the attacker. In this manner, the arcs on an avenue can be ranked by their ability to attain desired combat results.

In developing the attrition process for the combat simulation, an effort must be made to capture as accurately as possible the effects of the terrain the arc represents. Of great importance is the enemy's ability to deploy and the defender's ability to conduct counter-mobility operations. The algorithm must consider the deployment of the attacker forces from movement to assault formations. Limitations on assault formations that are imposed by the arc terrain must be modelled, because they affect the attacker's ability to conduct his attack. The object here is to accurately depict the force levels of the combatants to achieve the most realistic results from the subsequent attrition process.

2. Development of Flow Rate

The following mechanism is proposed to model the buildup of force levels that occur between defender and an attacker deploying from a line of march. This development assumes the defender is a US Mechanized Infantry Company/Team (CO/TM) and the attacker is the lead battalion of a Soviet Motorized Rifle Regiment. The defender's force value is assumed to remain constant during the battle, except for the casualties incurred (i.e., he will not be reinforced). The attrition process will be developed in Chapter IV. Furthermore, the defender is assumed to be able to bring to bear all the combat potential of his force. The attacker's condition is different. In attempting to deploy

his forces into assault formation, he is constrained by the cross-country flow capacity of the arc. Equation 1.1 is be modified to represent this process:

SUAFLR = (VDEP * WDAVL) * R / (LDOC * WDOC) (eqn 3.1)

where:

SUAFLR = The unconstrained flow rate of attacker on this arc (SUA/Hr).

VDEP = Velocity of deployed attacker (Km/Hr).

WDAVL = Width available for assault deployment on arc (Km).

R = Battalion to SUA conversion factor (SUA/Bn, surrogate for GVS).

LDOC = Doctrinal length of deployed battalion (Km).

WDOC = Doctrinal width of deployed battalion (Km/Bn).

This equation allows enemy doctrine and composition, and terrain effects to be modelled in determining the value of the attacker force available for battle. WDAVL may be determined from arc attributes already on the network developed by Craig. LDOC and WDOC are available in intelligence literature. VDEP is also a function of threat doctrine and composition and may be bounded by arc terrain. An estimate for the conversion factor R can be made until the GVS research is available.

3. Obstacle Impact on Flow Rate

There is a major flow rate constraint that is not represented in Equation 3.1 : the impact of defender counter-mobility operations on the arc. The emplacement of mines, ditches, and other obstacles add delay time to the arc and reduce the attacker's deployment flow rate. The impact of obstacle delay on flow rate is described by:

TOTFLR = SUAFLR / (60 + ARCDLY) (eqn 3.2)

where:

TOTFLR = Total attacker force flow rate (SUA/Min).

SUAFLR = Unconstrained arc flow from Eqn. 3.1 (SUA/Hr).

ARCDLY = Total number of minutes of delay time added by the defender's obstacles now on the arc plus the effects of the next best, feasible obstacle.

This representation gives an obstacle a delay time valid for the duration of the battle. In actual combat an obstacle's time delay value would decrease over time as the enemy reduced the obstacle. It is maintained as a constant for planning purposes.

4. Determination of Force Levels

The value of attacker forces that has been available to participate in combat is then TOTFLR multiplied by the time elapsed since the start of combat. Of course, this value has an upper bound of the amount of enemy forces initially available to attack. The attacker cannot deploy more forces than he has. Deployment may be regulated by determining the amount of time required to deploy the entire attacking force and using it as part of a time-stepped attrition process.

$$\text{TMSTP1} = \text{ATRTOT}/(\text{TOTFLR} * \text{DELTAT}) \qquad \text{(eqn 3.3)}$$

where:

TMSTP1 = Number of time-steps required to deploy the entire attacking force on the arc (time-steps).

ATRTOT = Total value of the attacker (SUA).

TOTFLR = Total flow rate of the attacker deploying on the arc (SUA/Min).

DELTAT = Desired duration of time-steps for attrition process (Min/Step).

The foregoing developments now permit the calculation of force levels representative of those that would be found in combat on a particular arc at a given time(t). For the defender:

$$DEFFOR(t) = DEFFOR(t-1) - DEFATR(t-1) \qquad \text{(eqn 3.4)}$$

where:

DEFFOR(t) = Value of defender forces at the start of time-step(t) (SUA).

DEFFOR(t-1) = Value of defender at the start of the time-step(t-1) (SUA).

DEFATR(t-1) = Value of attrition the defender encountered during time-step(t-1) (SUA).

As mentioned earlier, for these initial evaluations of arc feasibility, the defender is assumed not to receive reinforcements. The force level of the attacker at the beginning of time-step(t) is determined by:

$$ATFOR(t) = ATFOR(t-1) - ATATR(t-1) + (TOTFLR * DELTAT) \qquad \text{(eqn 3.5)}$$

where:

ATFOR(t) = Value of attacker at the start of time-step(t) (SUA).

ATFOR(t-1) = Value of attacker at the start of the previous time-step(t-1) (SUA).

ATATR(t-1) = Value of the attrition the attacker suffered during time-step(t-1) (SUA).

TOTFLR = Total of flow rate of the attacker deploying on the arc (SUA/min).

DELTAT = Minutes in current time-step (min/step).

The net result of the preceding equations is a method to determine the approximate buildup of forces between defender and an attacker deploying from march column. The effects of doctrine, force composition, and terrain are considered. These features will permit a more accurate attrition process.

5. Obstacle Selection Process

As mentioned above, one of the benefits of this methodology is the ability to represent the effects of defender counter-mobility operations and terrain potential. The term ARCDLY in Equation 3.2 serves to reduce the flow rate of the attacking forces by the effects of defender counter-mobility operations. This term will now be developed.

When evaluating an arc from a counter-mobility standpoint, two things must be considered. First, the effects of previously conducted countermobility operations must be measured. The delay imposed by an obstacle will depend to a large degree on the enemy's composition, engineer resources, and the defender's coverage of the obstacle by fire. As shown by Kazimer [Ref. 3: p. 46], these times can be determined. Thus, initially:

$$\overline{ARCDLY} = \sum_{i=1}^{n} OBSTDLY(i) \qquad \text{(eqn 3.6)}$$

where:

OBSTDLY (i) = the delay time imposed on the arc by the ith obstacle on that arc.

Secondly, the potential of the arc for future counter-mobility operations must be considered. All other conditions being equal, the terrain that allows the enemy to be delayed more will be the better battle position. To determine this potential, the terrain's counter-mobility capacity must be evaluated along with the defender's ability to conduct the contemplated counter-mobility operations. An algorithm to accomplish this sequential process was implemented by Craig [Ref. 1: pp. 79-83] and will be used in the present research, with minor modifications.

To assess terrain countermobility capacity, Craig employed the Engineer Standard Operating Procedure Table

Array shown in Table I . For each type of arc in the network, a yes/no (1/0) entry is made in the matrix on the left side of the array to indicate the ability of the arc to be effected by a particular type of obstacle.

The resources required to implement a particular obstacle are indicated in the right hand portion of the array. On the extreme right of the table, the columns "cost" and "delay" indicate the relative value of the assets required to complete a given obstacle as compared to the value of the assets in all other obstacles and the delay time that may be expected to be imposed on a Soviet Motorized Rifle Regiment. Differences in cost and delay time between obstacle types are caused by differences in the standard asset package quantity required for construction. This work is an extension of Kazimer [Ref. 3: p. 43].

An algorithm to determine the counter-mobility potential of every arc on the optimal avenue of approach and represent this with the best feasible obstacle follows:

Inputs: An avenue of approach, composed of arcs and nodes with the attributes listed in Appendix A, through a given sector; an Engineer SOP Table Array; and a set of engineer resources available for counter-mobility operations.

Outputs: The obstacle for each arc that will impose the most time delay on the attacker and is feasible with the engineer resources currently available.

Step 1. For every arc in the avenue of approach, define a set of possible obstacle types from the Engineer SOP Table Array. If every arc has been checked, stop. If no obstacles are possible for an arc, go to Step 4.

Step 2. For every obstacle type in the set defined in Step 1, beginning with the most delay producing obstacle and proceeding sequentially to the least delay producing obstacle, find the first obstacle for which resources are available to construct it. If no obstacles are feasible, go to Step 4.

Step 3. Assign the obstacle and delay time found in Step 2 to the arc for determining the potential impact of counter-mobility operations on the enemy's deployment flow rate on this arc. Return to Step 1.

Step 4. Assign neither obstacle nor delay time potential to this arc. Return to Step 1.

TABLE I

ENGINEER SOP TABLE

	ARC TERRAIN CODE										STANDARD ASSET PACKAGE QUANTITY							COST	DELAY
OBSTACLE TYPE	1	2	3	4	5	6	7	8	9	10	Squads	Dozers	Brdg Dem	M180	MFJ	MOPMS	Fuel		
Blow Bridge	0	0	0	0	0	0	0	0	0	1	1	0	4	0	0	0	0	1	1.5
Block Primary Rd (100m)	1	0	1	0	0	0	0	0	0	0	2	0	0	2	0	0	0	2	.5
Block Secondary Rd	0	0	0	1	1	1	0	0	0	0	1	0	0	1	0	0	0	1	.25
Block Secondary Rd	0	0	0	1	1	1	0	0	0	0	1	0	0	1	0	1	0	2	.75
Block Secondary Rd (100m)	0	0	0	1	1	1	0	0	0	0	1	0	0	1	1	1	0	4	.75
Block Open Field (300m)	0	0	0	0	0	0	0	1	0	0	3	0	0	0	3	0	0	3	.75
Block Open Field (300m)	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	3	3	.1

In this algorithm, only one obstacle is drawn from the set of applicable and feasible obstacles defined for an arc. This number was chosen because it was felt that, with some engineer resources available to the defending battalion, each company position could reasonably expect to receive at least one counter-mobility operation. If engineer resources were at a very high level, this algorithm could be modified to determine the best two or more feasible obstacles. This process could also be used to improve the defense in the event the defender breaks before the attacker.

The calculation of ARCDLY in Equation 3.6 is now modified to include the value of potential counter-mobility operations on an arc:

$$\text{TOTDLY} = \sum_{i=1}^{n} \text{OBSTDLY}(i) + \text{POTENOBST} \qquad \text{(eqn 3.7)}$$

where:

TOTDLY = The time delay imposed on an arc by all obstacles currently emplaced on the arc and the time that would be added by the construction of the next best, feasible obstacle (Min).

OBSTDLY(i) = Delay time imposed on the arc by the ith obstacle on that arc (Min).

POTENTOBST = Delay time that would be imposed by the construction of the best feasible obstacle for that arc (Min).

ARCDLY will now allow the TOTFLR computed in Equation 3.2 to model the constraints on force buildup imposed by current and potential obstacles. This will permit more accurate representation of terrain effects in the attrition process.

This section developed the concepts of flow rate and arc obstacle delay to provide accurate force levels for a time-stepped attrition process. The attrition equations for that process will be developed in Chapter IV. Casualties are considered to come primarily from three potential

sources: direct fire, indirect fire, and close air support (CAS). Equations for other sources of casualties, such as nuclear, biological, and chemical (NBC) weapons, can be developed as required.

6. Determination of Battle Duration

Knowing the force levels at each time-step and the attrition equations to be used, the last component needed for the attrition process to be defined is battle duration. The arc battle will terminate when the one of three conditions is met: 1) the defender reaches breakpoint; 2) the attacker reaches breakpoint; or 3) the attacker moves past the defender.

The first two critieria can be checked easily. The third criteria requires the determination of the time it will take the attacker to close with the defender. Among the numerous factors that come into play here are the initial distance between the defender and the attacker, speed of the attacker's assault, and the effect of casualties on the attacker's movement toward's the defender.

Finding the distance between attacker and defender requires an estimation of their locations. Since an arc's attributes are constant over the length of the arc, the defender has no advantage for being at one point of the arc over another. The defender is therefore assumed to be initially located at the midpoint of the arc. In the absence of detailed line of sight information, the direct fire battle is assumed to begin when the attacker reaches the source node of the defender's arc. The attacker begins his deployment at this time. The distance between the two is then simply half the arc length. The closure time between the two forces would normally be this distance divided by the deployed speed of the attacker. However, attrition and the effects of being fired upon will cost the attacker additional time. The time required for the

attacker to deploy his forces on the arc will be used as an upper bound for this additional time. Depending upon the type of arcs picked by the avenue of approach algorithm, attacker deployment times have been about two to three times larger than the time required for him to move from the source node of the arc to arc midpoint. By using the sum of these two times as the estimated battle duration, the deployed speed of the attacker is effectively reduced by 25 to 35 percent. Thus, a method to determine the upper bound for the duration of the arc battle has been established.

```
BATLTM = TMSTP1 + (((ARCLNG/2) * 60)            (eqn 3.8)
         / (VDEP * DELTAT))
```

where:

BATLTM = Number of time-steps of battle duration.

TMSTP1 = Time-steps required to deploy attacker (Eqn. 3.3).

ARCLNG = Arc length (Km).

VDEP = Attacker deployed velocity (Km/hr).

DELTAT = Length of time-step (Min/Step).

With the development of the components of the attrition process (force levels, equations, and stopping rules), the arc battle can be fought. Its duration will provide part of the time being optimized. At its completion one has total attacker and defender casualties. Since the purpose was to evaluate arcs on their ability to impart delay to the enemy as a result of attrition, a mechanism must now be developed to convert attrition into time delay. This development is based on the premise that every casualty inflicted on an attacker delays its subsequent movement due to the time needed to treat the casualty, and to reorganize and consolidate forces for continuning the attack. For testing purposes here, this delay time is initially estimated at two and a half minutes for every one percent of the total attacking battalion that becomes a casualty. Thus, if after

an arc battle the attacker had received ten percent casualties, he would be delayed twenty five minutes before being able to resume the attack. The twentyfive minute delay time caused by battle casualties is added to the duration time of the battle.

The combination of these two times is evaluated when determining which arc on the avenue to select for unit placement. This procedure would be followed when the battle terminated due to defender reaching breakpoint or expiration of time. If the battle ends as a result of the attacker reaching breakpoint, the arc delay time is assigned a very large number, in this case 999, to represent the unit's inability to continue the attack until major replacements are received. The total delay time created by the placement of a defender on an arc is useful when attempting to determine the attacker's traversal time on the expected avenue of approach. As Craig points out [Ref. 1: p. 63], the traversal time of an arc is equal to the time required to travel over the arc plus the time required to fight through a defender on the arc. This would enable the Planning Model to calculate how long the defender could expect to delay an attacker and determine if this plan satisfied mission requirements.

7. <u>Weighting Scheme to Favor FLOT Positions</u>

As stated earlier, the defender would generally want to break the attacker or, barring that, inflict maximum travel time delay. This could very well lead to the optimal arc being found far to the sector rear. In fact, the results of Chapter V will demonstrate this. Very seldom will commanders have the lattitude to defend initially in the rear of their sectors. Other factors may require him to defend close to the FLOT. A weighting scheme is therefore needed in the arc evaluation process so that proximity to the FLOT as well as casualty inflicting potential is considered. The general nature of this function would resemble:

ARCVAL = CASDLY * FLOTFN (eqn 3.9)

where:

ARCVAL = The value of the arc for comparision purposes with other arcs on the avenue of approach (min).

CASDLY = The time delay produced by the attacker's casualties during the arc battle (min).

FLOTFN = The value of the current arc's location as determined by a weighting function, ranging from One to Two. This range is chosen instead of Zero to One because it provides a linear weight and allows the last arc in the path to be eligible for unit placement; as would be the case where it is the only arc that allows the defender to break the attacker.

One of the first issues to be addressed in developing a weighting function is the nature of the form of the function. The curve might be linear with a slope of one; indicating that being midway between the FLOT and the rear boundary is half as good as being on the FLOT. Alternatively, the curve might be exponential, indicating arcs close to the FLOT are more heavily weighted than those to the rear. The development that follows will assume a linear curve with a slope of one is more applicable in the general case. The weighting function curve shape will be input to the model. See Figure 3.1 for the discussion that follows:

Step 1. Determine the slope of the line segment connecting the two FLOT endpoints, segment AB.

Step 2. Determine the midpoint of the last arc on the avenue of approach one that crosses that sector rear boundary, point C. Construct line segment CH with a slope equal to that of line segment AB.

Step 3. Determine the midpoints of the line segments connecting the two sector rear boundary points and the two FLOT endpoints, points E and F.

Step 4. Determine the slope of line segment EF.

Step 5. Determine the point of intersection between line segment CH and the line formed at the midpoint of the first arc on the avenue of approach to cross the FLOT with a slope equal to line segment EF. This intersection defines the line segment DG.

Step 6. Determine the length of line segment DG.

Step 7. For every arc on the avenue of approach, do:

a. Determine the midpoint of the arc.
b. Construct a line from the point found in Step 7a with a slope equal to that of line segment EF and extend it to the point of intersection with line D.
c. Determine the length found in Step 7b.
d. Divide the length found in Step 7c by the length found in Step 6. Add one to this fraction. The resulting number, ranging from one to two, is the FLOTFN of Equation 3.9
e. Solve Equation 3.9 for ARCVAL and store result. Loop.

Step 8. Select the arc with the largest value of ARCVAL on the current avenue of approach for unit placement.

The result of this process is a linear weighting function where the arc closest to the FLOT receives more credit for its arc battle delay time and arcs closer to the sector rear receive proportionally less.

8. Covering Multiple Arcs with One Unit

The last portion of this section will address the condition Craig noted in his initial work with the terrain network [Ref. 1: p. 72]. It was discovered that in the process of placing units on the network and recalculating the avenue of approach, several units would often be placed around a single node. While such a disposition might occur, normally a commander would first consider the possibility of controlling these arcs by placing a unit at the node of intersection. A portion of the defender's value would then be allocated to each arc entering the node to represent the unit's ability to cover multiple avenues of approach. By adding a portion of the defender's SUA value to each arc, the likelihood of an adjacent arc being selected again is greatly reduced unless it truly is the best arc available.

This problem of overallocation can be solved by using the presence of defender forces on an arc to initiate several checks. The process is shown schematically in Figure 3.2 and outlined below:

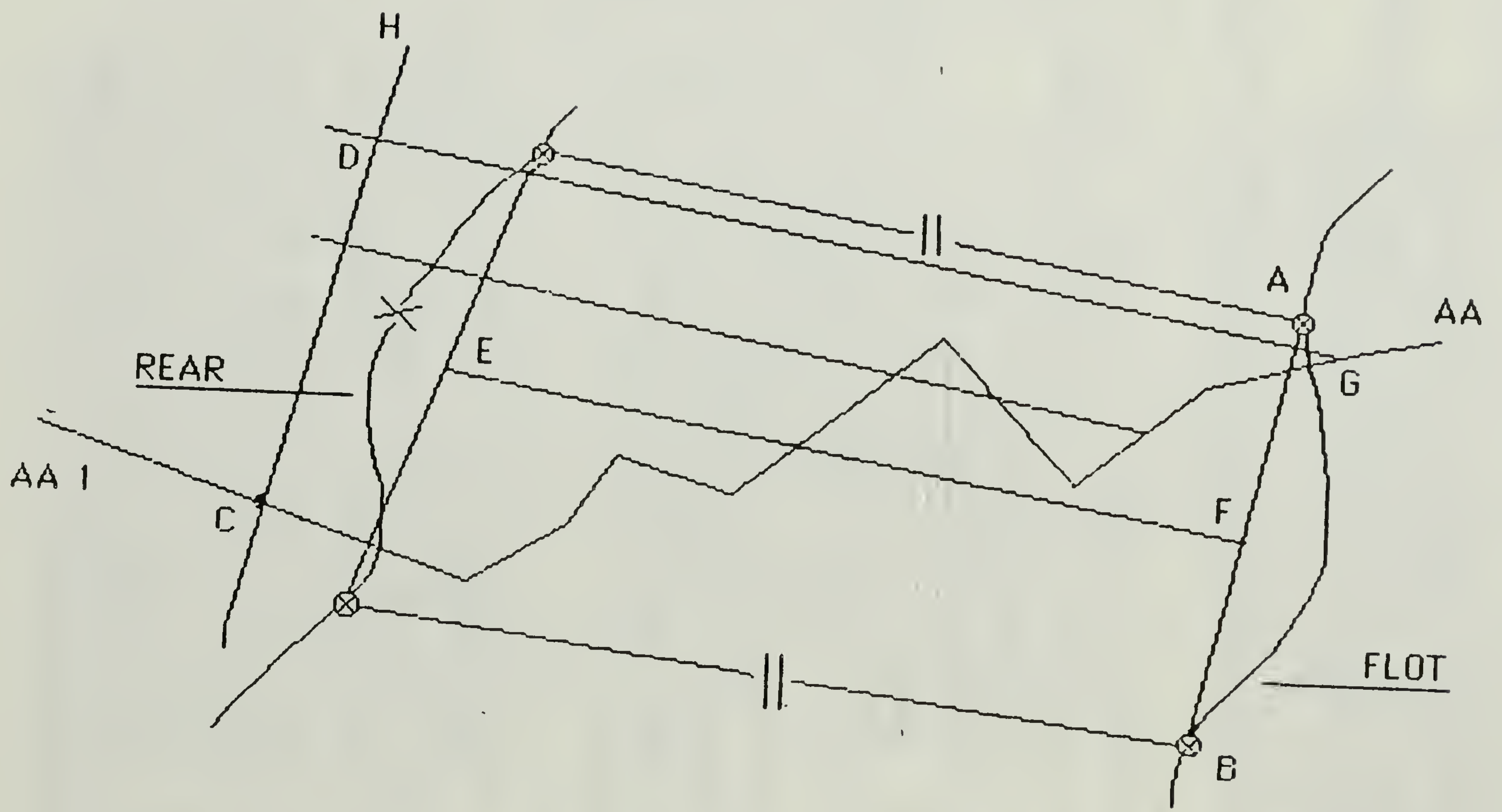


Figure 3.1   Weighting Scheme for Arc Selection.

Inputs: A candidate arc that has been selected to receive a unit based upon the results of the arc evaluation process, arcs and nodes with the attributes listed in Appendix A, and a unit being considered for placement on the terrain network.

Outputs: A decision whether to assign the unit to the candidate arc or locate a unit on an adjacent arc on the common node.

Step 1. Determine if either end node of the candidate arc indicates the presence of a unit on an arc adjacent to the node.

Step 2. If neither node indicates the presence of a unit on an adjacent arc, go to Step 8.

Step 3. If both end nodes indicate the presence of a unit on an adjacent arc, then search the candidate arc for the presence of a unit on the arc. If the candidate arc has a force on it, go to Step 8.

Step 4. Search every arc entering the source and sink nodes of the candidate are to determine the arc adjacent to either node that has the largest valued force on it.

Step 5. Remove half of the value of the force found in Step 4 from its current arc and add that amount to every other arc that enters the end node to which it was adjacent. Change the ownership of the unit that was distributed around a node from its arc to the node around which its value was distributed. Go to Step 9.

Step 6. If either end node of the candidate arc indicates the presence of an adjacent arc occupied by a force, then search every arc entering that node for the force with the largest value.

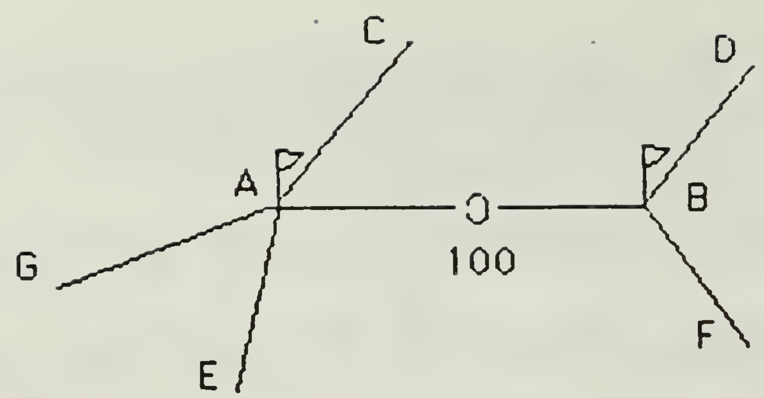
Step 7. Remove half of the value of the force found in Step 6 from its current arc and add that amount to every other arc that enters the node. Change the ownership of the unit that was distributed around a node from its arc to the node around which its value was distributed. Go to Step 9.

Step 8. Place the unit on the candidate arc. Set the attribute of the end nodes of the candidate arc to indicate the presence of a unit on the arc. Go to Step 10.

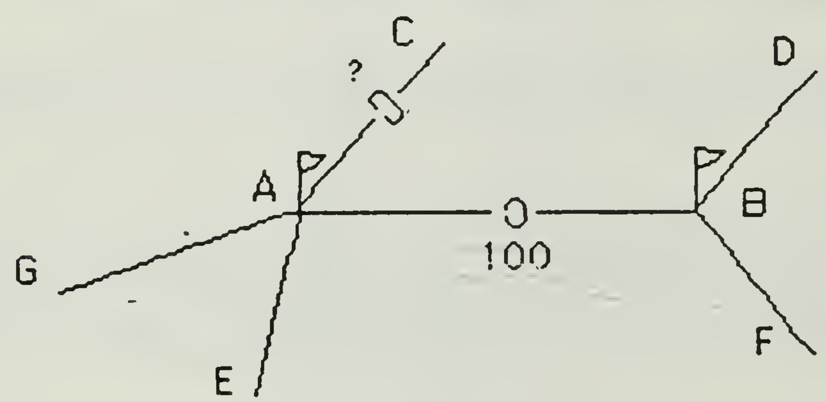
Step 9. If no other units are owned by the arc from which the force value was subtracted, and no units exist on any arc entering either end node of this arc, then set the end node attribute of that arc to indicate no presence of a unit.

Step 10. Stop.

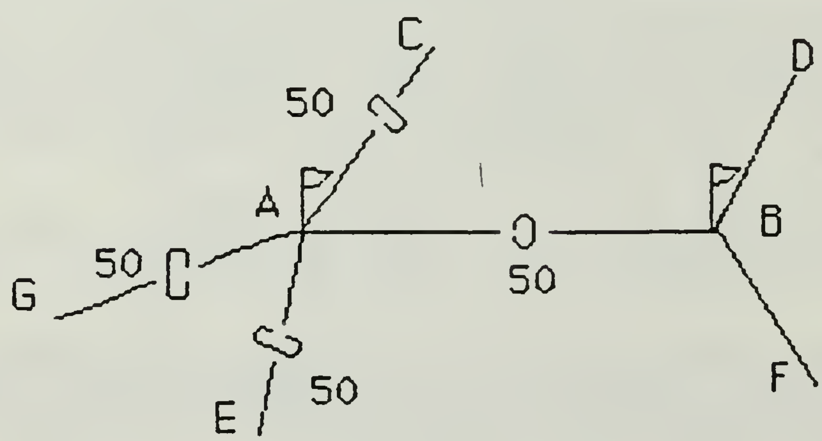
Once a unit has been moved from an arc to a node, it is not considered eligible for further movement, because it is covering at least two critical avenues of approach. It is possible for more than one unit to be moved to a node,



Unit of value 100 placed on arc AB. Flags set at end nodes indicate unit on adjacent arc.



Arc CA on avenue C-A-E identified as optimal arc. Presence of flag results in search and unit relocation. Flags removed.



Distribution of half of force around node A ensures full value of unit is considered on subsequent avenue searches through A. Force value would be consolidated during execution to avoid duplication.

Figure 3.2 Method to Cover Multiple Arcs With One Unit.

but this has not occurred in testing to date. If a unit is distributed around a node, then new obstacles will be placed on those arcs nearer to the FLOT. If the unit's initial obstacle is behind it as a result of being moved, then the obstacle will be removed from the network.

The arc selection process considers combat only on the arc being evaluated, i.e., adjacent arcs are not considered. If the adjacent arcs are found to be significant (in that they are candidates for unit placement on subsequent passes) the process described above captures the interrelationship of the arcs. Otherwise, the adjacent arc is not a factor in the Planning Model. Units will in fact engage in combat over multiple arcs, when targets can be acquired, in the Execution Model.

This section has developed two methods for evaluating arcs on an optimal avenue of approach for unit placement. The first method generates the most delay time due to attrition of the attacker. The second uses a weighting function applied to the delay time generated in the first method to give preference to a position closer to the FLOT. Both methods incorporate a time stepped attrition process in which the effects of arc terrain flow rates and countermobility operations are considered. The concept of flow rates allows for simple yet effective consideration of arc terrain and attacker doctrine and composition. These methods permit arc evaluation on the basis of combat simulation results as compared to previous evaluation schemes that ranked arcs on the basis of one physical attribute. Additionally, this section has developed a method to correct the problem of over allocating units around a node that was noted in earlier research. The results of the algorithm to correct overallocation are noted in Chapter V.

B. AVENUE OF APPROACH OPTIMIZATION

As discussed in the Chapter I, previous sector avenue of approach generation methods relied on finding a minimum travel time path. While applicable in some situations, the discussion in Chapter II showed that wide variation exists in the type of avenue that may be needed.

1. MAX FLOW Avenue of Approach

Three avenue of approach generation algorithms are proposed to cover the more common tactical scenarios. These algorithms will define the arc cost for the Dykstra minimum path algorithm currently used by the model. The first algorithm, named MAX_FLOW, attempts to find the path that provides the highest flow rate through the sector. The tactical significance of this algorithm would be the determination of the path through the sector that provides the best ability to deploy and manuever off the road. This algorithm would be helpful to an attacker who needed to pass through heavily forested or restricted terrain and was concerned with delays that might arise from ambushes or chokepoints. The algorithm is described below.

Inputs = Terrain network containing arcs and nodes with the attributes listed in Appendix A with defender's sector defined; VDEP, deployment velocity of attacker; R, battalion to SUA conversion factor; LDOC,doctrinal length of deployed attacker battalion; WDOC, doctrinal width of attacking battalion; ATRTOT, total value of attacker.

Outputs = COST, the cost of an arc, which will be passed to the minimum path solution routine for every arc in sector.

For every arc in the defender's sector :

Step 1. Determine WDAVL, the width available on the arc for assault deployments of the expected attacking force. The width is determined by considering the off-road width, the type of unit the off-road terrain will suppport, and the number off-road lanes on the arc. All of these inputs are arc attributes.

Step 2. Calculate the unconstrained flow rate for the arc by solving Equation 3.1

Step 3. Calculate the total flow rate of the arc, incorporating the added delay of any obstacles emplaced on the arc by solving:

$$ARCFLR = SUAFLR \,/\, (60 + \sum_{i=1}^{n} OBSTDLY(i)) \qquad \text{(eqn 3.10)}$$

Equation 3.10 closely resembles Equation 3.6, the solution for TOTFLR. The difference is that Equation 3.10 accounts only for the actual obstacles emplaced on the arc, while Equation 3.6 includes one potential obstacle's delay. Equation 3.10 credits the enemy with knowing what obstacles are in place in the defender's sector but does not allow him to consider the defender's countermobility potential.

Step 4. Divide ATRTOT by ARCFLR to determine number of minutes required to deploy attacker on the arc:

$$TMSTP2 = ATRTOT \,/\, ARCFLR \qquad \text{(eqn 3.11)}$$

Step 5. Let COST = TMSTP2. Save the value of the arc's cost. If all arcs in sector have not been analyzed, Loop.

Step 6. Return all values of arc COST to the minimum path solution routine. Stop.

The minimum path produced by this algorithm would be that avenue of approach passing through the sector that had the smallest value of the summed costs of the linked arcs. Since the smallest cost is produced by the highest flow rate, the arc with the best flow rate and best off-road manueverability is selected.

2. MIN_TIME Avenue of Approach

The second algorithm to be developed is named MIN_TIME. As the name implies, it solves arc cost in terms of the minimum travel time for an attacker across an arc. For simplicity, the attacker force is treated as a point for all arc computations and travel time is the arc length divided by average road speed of the attacking force. The former is an arc attribute, while the latter is determined from unit movement capabilities. The algorithm is as follows:

Input: Terrain network containing arcs and nodes with attributes listed in Appendix A with defender's sector defined; ARCLNG, an arc attribute; VTRVL, road travel speed of attacker; VBRUSH, speed of attacker clearing a path through a heavily vegetated arc with no roads; VOPEN, speed of attacker travelling across open terrain. DEFONARC, an arc attribute that allows determination if a defender has previously been assigned to the arc being evaluated.

Output: COST, the cost of an arc, which will be passed to the minimum path solution routine for every arc in sector.

For every arc in the defender's sector:

Step 1. Determine which Speed, VTRVL, VBRUSH, or VOPEN, applies to the arc.

Step 2. Let COST = ARCLNG/Speed from Step 1.

Step 3. If DEFONARC = yes, Let COST = 999. Otherwise go to Step 4. This step allows the effect of placing a defender in the sector to be modelled. It allows the next quickest route to be found once a defender has been placed on the minimum time path.

Step 4. Save the value of COST. If all arcs in sector have not been analyzed, Loop.

Step 5. Return all values of arc COST to the minimum path solution routine. Stop.

This algorithm finds the quickest avenue of approach through the sector. Any type of arc can be on this avenue and speeds used to determine travel time are adjusted for terrain. An avenue of this nature might be used by raiding or reconnaisance elements of the attacker.

3. BEST ROAD Avenue of Approach

The third algorithm, named BEST_ROAD, attempts to find the avenue with the best road surfaces. Arcs are evaluated on their ability to sustain heavy traffic. This algorithm might be used by an attacker with significant logistic requirements to determine his best route from a sustainability standpoint. A subjective weight is assigned to each type of arc surface. These weights are derived from the author's experience in command and staff positions in Mechanized Infantry Units over a period of three years. The arc's weight is multiplied by the reciprocal of the number

of lanes on this arc times the length of the arc. By assigning the lowest weight to the best surface, the best arc receives the lowest cost and is optimized in the minimum path routine. The algorithm is as follows:

Input: Terrain network containing arcs and nodes with attributes listed in Appendix A and defender's sector defined; RTECLAS, arc attribute describing surface type; LANENO, arc attribute describing number of lanes on the arc; DEFONARC, an arc attribute that allows determination if a defender has previously been assigned to the arc being evaluated; and ARCLNG, an arc attribute that the specifies arc length. Additionally, the COST coefficients have been provided here but would normally be input to reflect varying local conditions.

Output: COST, the cost of an arc, which will be passed to the minimum path solution routine for every arc in sector.

For every arc in the defender's sector:

Step 1. Determine RTECLAS and LANENO of the arc and find COST:

```
If RTECLAS = Highway
    Let COST = 1 / (LANENO * ARCLNG)
If RTECLAS = Highway and railroad
    Let COST = .8 / (LANENO * ARCLNG)
If RTECLAS = Railroad
    Let COST = 4 / (LANENO * ARCLNG)
If RIECLAS = Concrete road
    Let COST = 2 / (LANENO * ARCLNG)
If RTECLAS = Asphalt road
    Let COST = 3 / (LANENO * ARCLNG)
If RTECLAS = Dirt road
    Let COST = 5 / (LANENO * ARCLNG)
If RTECLAS = Forest
    Let COST = 20 / ARCLNG
If RTECLAS = Open country
    Let COST = 8 / ARCLNG
If RTECLAS = Road and railroad

    Let COST = 2 / (LANENO * ARCLNG)
If RTECLAS = Bridge or Tunnel
    Let COST = 1.5 / (LANENO * ARCLNG)
If DEFONARC = yes
    Let COST = 999
```

Step 2. Save COST. If all arcs have not been evaluated, Loop.

Step 3. Return all values of arc COST to the minimum path solution routine. Stop.

The algorithm permits forests and open country to be evaluated; certain sectors may have this type of terrain predominate. Also, the presence of a defender increases the cost of an arc. This allows the next best path to be found once a unit has been placed in the sector.

This section has developed three algorithms which cover the more common attacker objectives in formulating an avenue of approach. In MAX_FLOW, the tactical significance of the terrain is considered in evaluating the arc. In MIN_TIME, the fastest route, regardless of logistic potential, is selected. In BEST_ROAD, the avenue with the best potential for follow-on logistic operations is determined. More algorithms can be developed for other attacker avenue objectives. Having developed these alternatives, the Planning Model must pick the appropriate algorithm. This would be a function of input from higher echelon and intelligence networks, which are areas of future research. The results of these algorithms are examined in Chapter V.

IV. MODEL COMPONENTS

A. DIRECT FIRE ATTRITION

As mentioned in the previous chapter, the attrition process used in the present algorithm has three components representing the effects of direct fire, indirect fire, and CAS. The equations used to determine these components are developed in this chapter. The first equation to be examined is that which describes the direct fire attrition. This equation seeks to model the effects of the primary direct fire weapons of the ground combatants: small arms, tank, and armored personnel carrier fire.

Numerous candidate equations for this type of process have been developed. The one used here is a form of the equation for Helmboldt-type combat as described by Taylor [Ref. 6: p. 36]. The Helmboldt Equation has been used for several reasons. Helmboldt's hypothesis that the larger force suffers inefficiencies of scale in inflicting casualties is compatible with the earlier objective of using flow rates to determine force levels during the attrition process. It is an effort to represent aspects of the battle process that are not apparent in considering overall force sizes and normal Lanchesterian outcomes, but which nonetheless occur and have an important impact on the battle. Additionally, the flexibility provided by the use of a power function to represent the fire-effectiveness modifications caused by the relative force ratios is very helpful in a research context.

A form of Helmboldt's Equation in which the attrition coefficients are not time-dependent and the modification factor is a power function incorporating the Weiss parameter as described by Taylor [Ref. 6: p.38] will be used here:

$$dx/dt = -a * y * ((x/y)^{**}(1 - W)), \quad x(0) = x0 \quad \text{(eqn 4.1)}$$

$$dy/dt = -b * x * ((y/x)^{**}(1 - W)), \quad y(0) = y0 \quad \text{(eqn 4.2)}$$

As Taylor points out [Ref. 6: p. 39], by varying the value of the Weiss parameter from 1 to 1/2 to 0, one can alternately realize the square, linear (approximate), or logarithmic forms of Lanchester's Equation. Chapter VI examines the impact of several values of the Weiss parameter on the direct fire attrition.

B. INDIRECT FIRE ATTRITION

In developing an attrition process for indirect fire, the goals used with direct fire still apply. That is, a process is desired that captures the effects of doctrine, terrain and composition. In the algorithm presented here, artillery is employed when the attacker and defender fight on the same arc. The ability of one side to engage the other at distant ranges is not represented. Ongoing research on the GVS and artillery fire planning will address long range fires. See the works of Kilmer, Lindstrom, and Finley [Refs. 7,8,9]. Of greater interest in the present instance is the effect of the artillery on the enemy's assault on the current arc.

To accomplish the forementioned objective, the attacker force value, at a given time-step in which artillery is employed against him, is divided by the product of his assault echelon's doctrinal length multiplied by the available off-road deployment width. This produces an attacker force/terrain density measured in SUA/Km^2. This density is then multiplied by the casualty area (Km^2) of the defender's firing unit, the casualty area produced by firing one round per gun in a battery. The product of the area times the density is the value of the attacker lost to one salvo. This loss is then further multiplied by the number of salvos

the defender fires per minute times the number of minutes per time step. In the description below, DCEP refers to an area. The DIFCAS variable determines the percentageof casualties within the DCEP area. An equation for attacker attrition from indirect fire is:

$$ARTAT = (ATRFOR / (LDOC * WDAVL)) * DCEP * DIFCAS * DROF * DELTAT \qquad \text{(eqn 4.3)}$$

where:

ARTAT = Attacker attrition from artillery during the timestep (SUA/Step).

ATRFOR = Value of attacker currently on arc (SUA).

LDOC = Doctrinal length of deployed echelon (Km).

WDAVL = Width available on the arc for off-road deployment to the attacker (Km).

DCEP = The casualty area of one salvo of one of the defenders' indirect firing units (Km^2).

DROF = Defender artillery rate of fire (Salvos/minute).

DIFCAS = The percentage of casualties defender artillery will inflict on attacker forces in the DCEP area per firing unit salvo (Percent Casualties/Salvo).

DELTAT = Number of minutes in the current time step (Min/Step).

LDOC is used as an estimate for attacker formation length to simplify calcualations. Equation 4.3 is meant to represent the effects of area munitions such as high explosive and dual purpose improved conventional munition. It is not appropriate for munitions such as laser guided projectiles and scatterable minefields. It is recognized that formation lengths during an assault will vary from this over time as the attacker deploys from column to line.

The defender's attrition from indirect fire is modelled slightly differently. Having the benefit of selecting the terrain and preparing the battlefield, the defender is assumed to not face the same deployment constraints as the attacker. It would therefore not be correct to use the same

process as outlined in Equation 4.3 . Instead, a straight casualty percentage will be applied against the defender's value, multiplied successively by the attacker's rate of fire and the timestep duration.

$$DARTAT = DEFFOR * AIFCAS * AROF * DELTAT \qquad \text{(eqn 4.4)}$$

where:

DARTAT = Defender's attrition from artillery during the time-step (SUA/Step).

DEFFOR = Value of defender currently on arc (SUA).

AIFCAS = The percentage of casualties attacker artillery will inflict on the defender per firing unit salvo (Percentage Casualties/Salvo).

AROF = Attacker artillery rate of fire (Salvos/Min).

DELTAT = Number of minutes in current time-step (Min/Step).

C. ATTACK HELICOPTER SUPPORT

The third attrition process that will be modelled deals with casualties produced by attack helicopters. Attrition from other forms of CAS is an area of ongoing research. The purpose of providing this process is to enable the planning model to have access to the additional resources the defender may need to accomplish his mission. It is also available for inclusion in the attacker's order of battle when applicable. Normally, the defender will not initially use helicopters. The model will be able to draw upon them, if allocated, to see if their employment in an arc battle allows the defender to achieve his objective.

The attrition process for both defender and attacker helicopters is essentially the same. The defender's equation is slightly different to account for the US practice of rotating attack helicopters in thirds to the battle. One element is engaged, one is returning to rearm, and the third element is enroute to the battle. In this manner, attack

helicopters are constantly participating in the battle. The attrition process involves determining an expected casualty per minute of effective employment rate. A subjective evaluation is made by the author of the expected casualties an attack helicopter might inflict on high value targets such as tanks given a normal combat load of missiles, rockets, and cannon. This expected casualty production is then divided by the amount of time the helicopter would normally spend engaging the enemy. The result is a casualty per minute rate when actually engaging the enemy. With this casualty rate, the effective mission duration, and the number of helicopters involved, the helicopter attrition process can be added into the arc battle. In the current model the defender's helicopters are made available at the start of the arc battle, but this can be modified to simulate arrival later in the battle. The defender equation for helicopter attrition against the attacker is:

$$AHLCAS = (DFHEL/3) * DHCSRT * DELTAT \qquad \text{(eqn 4.5)}$$

where:

AHLCAS = Value of attacker casualties caused by defender attack helicopters in the current timestep (SUA).

DFHEL = Total number of attack helicopters employed by the defender (Units).

DHCSRT = Expected casualty production of defender attack helicopter per minute of effective engagement (SUA/min).

DELTAT = Number of minutes in the current timestep (Min/Step).

An equation similar to this may be used for the attacker. The purpose of Equation 4.5 is not to produce a high resolution simulation of attack helicopter effectiveness. Rather, it is meant to provide an estimate of the impact of providing attack helicopters on the arc battle.

It will provide the planning model with a method to determine where these critical assets might be required in the defensive battle.

V. RESULTS

A. AVENUES OF APPROACH GENERATION

This chapter provides analysis of the results of testing the algorithms presented earlier. The three avenue of approach algorithms will be considered first. A common 10 X 20 Km sector and the same attacker and defender force values were used in testing each algorithm. The terrain network is that developed by Craig [Ref. 1] and converted into a Simscript model by Rolands and Associates, Inc. In the tables that follow, arcs are presented as they were encountered moving from the FLOT to the rear of the sector along the avenue of approach. The ARC NO. column refers to the position of the arc along the path; the first arc is normally the arc just in front of the FLOT while the last arc is the one that crosses the rear boundary. ARC TYPE and NO LNS entries describe the road surface and number of lanes of that surface on the arc. OFF-RT WD describes in Km how much room for deployment is available on the arc, and includes the off-route width as well as the road surface width. OFF-RT CLASS describes the classification of the off-road area for deployment. (See Appendix B for a description of this evaluation scheme.) LENGTH is simply the length of the arc. The COST column is the value of the arc returned by an avenue of approach algorithm and is the number used in the Dykstra solution routine. FLOW RATE is the value of the flow capacity developed in Equation 3.2 . Finally, POINTER is an arc attribute that uniquely identifies the location of the arc in memory.

1. BEST_ROAD Algorithm Results

Table II shows the results of the BEST_ROAD algorithm. The initial path has a large proportion of 4-lane autobahn arcs with 50% of its 32 Km length consisting of

TABLE II

BEST_ROAD ALGORITHM RESULTS

FIRST AVENUE GENERATED:

ARC NO	ARC TYPE	NO. LNS	OFF-RT WD (KM)	OFF-RT CLASS	LENGTH (KM)	COST	FLOW RATE (SUA/MIN)	POINTER
1	AUTOBAHN	4	0.20	2	2.00	0.13	38.19	1285648
2	AUTOBAHN	4	0.50	1	1.40	0.18	90.28	1297424
3	AUTOBAHN	4	1.00	1	2.40	0.10	177.08	1305296
4	AUTOBAHN	4	0.10	2	2.00	0.13	20.83	1295888
5	AUTOBAHN	4	0.10	2	2.70	0.09	20.83	1295568
6	CONCRETE.RD	2	0.50	1	3.70	0.27	88.54	1309392
7	AUTOBAHN/RAIL	4	0.20	1	3.50	0.06	38.19	1310224
8	CONCRETE.RD	2	0.10	1	3.00	0.33	19.10	1325264
9	CONCRETE.RD	2	0.20	2	3.00	0.33	36.46	1327632
10	AUTOBAHN/RAIL	4	0.50	1	2.00	0.10	90.28	1326800
11	ROAD/RAILROAD	2	0.10	1	4.00	0.25	19.10	1328144
12	ROAD/RAILROAD	2	0.50	2	2.00	0.50	88.54	1337360
	TOTAL:				31.70	2.47		

*** FIRST DEFENDER UNIT PLACED ON ARC NO 4 ***

SECOND AVENUE GENERATED:

ARC NO	ARC TYPE	NO. LNS	OFF-RT WD (KM)	OFF-RT CLASS	LENGTH (KM)	COST	FLOW RATE (SUA/MIN)	POINTER
1	AUTOBAHN/RAIL	4	0.05	3	0.70	0.29	12.15	1264336
2	AUTOBAHN/RAIL	4	0.05	3	3.50	0.06	12.15	1264144
3	AUTOBAHN/RAIL	4	0.10	1	2.50	0.08	20.83	1283280
4	RAILROAD	2	0.50	2	4.10	0.49	88.54	1283088
5	AUTOBAHN/RAIL	4	0.20	1	3.50	0.06	38.19	1310224
6	CONCRETE.RD	2	0.10	1	3.00	0.33	19.10	1325264
7	CONCRETE.RD	2	0.20	2	3.00	0.33	36.46	1327632
8	AUTOBAHN/RAIL	4	0.50	1	2.00	0.10	90.28	1326800
9	ROAD/RAILROAD	2	0.10	1	4.00	0.25	19.10	1328144
10	ROAD/RAILROAD	2	0.50	2	2.00	0.50	88.54	1337360
	TOTAL:				28.3	2.49		

*** SECOND DEFENDER UNIT PLACED ON ARC NO 2 ***

this type of surface. This conforms to the goal of the algorithm. It has found the shortest path through the sector with the widest and best road surfaces. While a desirable path from a transportation and logistic standpoint has been found, this route has a tactical drawback in that over much of its course it has less that two hundred meters of off-road manueverability. Additionally, the need to change roads every few miles would create navigation difficulties for an attacker.

The second avenue of approach generated in Table II shows the effect of placing a unit on an arc of the first avenue. Using one of the arc selection algorithms that will be demonstrated later in this chapter, a US CO/TM was placed on the arc at pointer 1295888. As previously discussed, the arc with a defender is now made unattractive to the Dykstra solution routine of the model. The next best path is generated. It can be determined by comparing the arc pointers of the two avenues that the new avenue bypasses the occupied arc and later rejoins the first avenue. The first five arcs of the second avenue are different from those of the first avenue. Although the second avenue is shorter in length than the first path, it was less favorable because in bypassing the occupied autobahn arc, it selected the more costly railroad arc, arc number four. While approximately 3.5 kilometer shorter, the second path includes a trip along a unimproved railroad bed.

2. MIN TIME Algorithm Results

The second avenue of approach generation algorithm to be examined is the MIN_TIME method. A path generated by it under conditions identical to those in the previous example is displayed in Table III . As expected, this avenue is comparatively shorter than the other examples. Disregarding road surface, it takes the most direct route through the sector and is 41% shorter than the BEST_ROAD

TABLE III

MIN_TIME ALGORITHM RESULTS

FIRST AVENUE GENERATED

ARC NO	ARC TYPE	NO. LNS	OFF-RT WD (KM)	OFF-RT CLASS	LENGTH (KM)	COST	FLOW RATE (SUA/MIN)	POINTER
1	AUTOBAHN	4	0.20	2	2.00	0.10	38.19	1285648
2	AUTOBAHN	4	0.10	2	1.70	0.09	20.83	1297616
3	AUTOBAHN	4	0.50	1	1.00	0.05	90.28	1303056
4	AUTOBAHN	4	0.05	1	1.60	0.08	12.15	1305104
5	DIRT.ROAD	2	1.00	1	3.00	0.15	175.35	1315344
6	AUTOBAHN	4	0.40	1	2.20	0.11	72.92	1322512
7	ASPHALT.RD	2	0.30	2	1.50	0.08	53.82	1323728
8	DIRT.ROAD	2	0.50	2	3.20	0.16	88.54	1331728
9	CONCRETE.RD	2	0.50	2	1.40	0.07	88.54	1342160
10	CONCRETE.RD	2	0.10	1	1.00	0.05	19.10	1344208
	TOTAL:				18.60	0.94		

*** FIRST DEFENDER UNIT PLACED ON ARC NO. 2 ***

SECOND AVENUE GENERATED:

ARC NO.	ARC TYPE	NO. LNS	OFF-RT WD (KM)	OFF-RT CLASS	LENGTH (KM)	COST	FLOW RATE (SUA/MIN)	POINTER
1	AUTOBAHN	4	0.20	2	2.00	0.10	38.19	1285648
2	AUTOBAHN	4	0.50	1	1.40	0.07	90.28	1297424
3	AUTOBAHN	4	1.00	1	1.00	0.05	177.08	1296080
4	DIRT.ROAD	2	1.00	1	1.00	0.05	175.35	1296400
5	DIRT.ROAD	2	0.10	1	0.80	0.04	19.10	1305808
6	DIRT.ROAD	2	0.10	1	1.00	0.05	19.10	1306832
7	DIRT.ROAD	2	0.05	1	1.40	0.07	10.42	1312976
8	DIRT.ROAD	2	1.00	1	2.20	0.11	175.35	1313296
9	DIRT.ROAD	2	0.20	2	1.20	0.06	36.46	1324752
10	DIRT.ROAD	2	0.10	2	1.00	0.05	19.10	1329360
11	DIRT.ROAD	2	1.00	2	2.00	0.10	175.35	1330384
12	DIRT.ROAD	2	1.00	1	1.00	0.05	175.35	1335312
13	CONCRETE.RD	2	0.20	1	1.00	0.05	36.46	1340112
14	CONCRETE.RD	2	0.50	1	2.50	0.13	88.54	1341648
	TOTAL:				19.50	0.98		

*** SECOND DEFENDER UNIT PLACED ON ARC NO 1 ***

path. In this case, COST is simply arc length divided by one of three applicable travel speeds, VOPEN, VTRVL, or VBRUSH. A travel speed of 20 Kmph is assumed for all arcs except forest and open country types.

The avenue uses two lane dirt roads for a third of its length. While this may be feasible for small units in moderate weather, dirt roads would not be preferred for the supply routes of echelons above regiment. The algorithm does a good job of finding direct routes that might be used by reconnaisance or raiding elements. As was the case earlier upon placement of a unit on the initial avenue, the second avenue of approach generated avoids the arc with a defender on it and the second path in Table III constructs virtually a completely different avenue.

3. MAX_FLOW Algorithm Results

The last avenue of approach algorithm to be examined is the MAX_FLOW method. The results of this algorithm are displayed in Table IV . It will be recalled that this method selects arcs with high flow rates, which is a function of off-road deployment width and the presence of counter-mobility obstacles. As hoped for, most of the arcs on this path have a kilometer of off-route width to deploy on and only one arc has less than 500 meters. This route would be helpful to an attacker who enjoyed great numerical superiority and sought avenues where he could maximize the deployment of his forces in the assault.

While providing the desired manuever room and being only slightly longer than the BEST_ROAD path in Table II, this path has limitations. More than the other algorithms, it is navigationally very difficult, moving abruptly from autobahn to road to open country. Such a distorted path would hamper logistc operations. Some arcs lack roads altogether and their use would be very weather dependent. The second path in Table IV is only 100 meters longer than the

TABLE IV

MAX_FLOW ALGORITHM RESULTS

FIRST AVENUE GENERATED.

ARC NO.	ARC TYPE	NO. LNS	OFF-RT WD (KM)	OFF-RT CLASS	LENGTH (KM)	COST	FLOW RATE (SUA/MIN)	POINTER
1	AUTOBAHN	4	0.50	1	1.40	11.87	90.28	1296912
2	AUTOBAHN	4	1.00	1	2.40	3.53	177.08	1304784
3	DIRT. ROAD	2	1.00	1	1.30	6.58	175.35	1306128
4	DIRT. ROAD	2	0.50	1	3.20	5.29	88.54	1307152
5	CONCRETE. RD	2	0.50	1	3.70	4.58	88.54	1308880
6	AUTOBAHN/RAIL	4	0.20	1	3.50	11.22	38.19	1309712
7	DIRT. ROAD	2	1.00	1	3.00	2.85	175.35	1312976
8	DIRT. ROAD	2	1.00	1	2.20	3.89	175.35	1312784
9	OPEN COUNTRY	0	1.00	3	2.50	3.46	173.61	1324048
10	DIRT ROAD	2	1.00	2	3.00	2.85	175.35	1329680
11	DIRT. ROAD	2	0.50	2	3.20	5.29	88.54	1331216
12	OPEN. COUNTRY	0	1.00	2	2.20	3.93	173.61	1340944
13	CONCRETE. RD	2	0.50	1	3.00	5.65	88.54	1340112
	TOTAL:				34.60	70.99		

*** FIRST DEFENDER UNIT PLACED ON ARC NO 4 ***

SECOND AVENUE GENERATED:

ARC NO.	ARC TYPE	NO. LNS	OFF-RT WD (KM)	OFF-RT CLASS	LENGTH KM)	COST	FLOW RATE (SUA/MIN)	POINTER
1	AUTOBAHN	4	1.00	2	1.00	8.47	177.08	1263120
2	OPEN. COUNTRY	0	1.00	2	1.00	8.64	173.61	1263312
3	AUTOBAHN/RAIL	4	0.05	3	3.50	35.27	12.15	1263632
4	AUTOBAHN/RAIL	4	0.10	1	2.50	28.80	20.83	1282768
5	RAILROAD	2	0.50	2	4.10	4.13	88.54	1282576
6	AUTOBAHN/RAIL	4	0.20	1	3.50	11.22	38.19	1309712
7	DIRT ROAD	2	1.00	1	3.00	2.85	175.35	1312976
8	DIRT. ROAD	2	1.00	1	2.20	3.89	175.35	1312784
9	OPEN COUNTRY	0	1.00	3	2.50	3.46	173.61	1324048
10	DIRT ROAD	2	1.00	2	3.00	2.85	175.35	1329680
11	DIRT. ROAD	2	0.50	2	3.20	5.29	88.54	1331216
12	OPEN COUNTRY	0	1.00	2	2.20	3.93	173.61	1340944
13	CONCRETE. RD	2	0.50	1	3.00	5.65	88.54	1340112
	TOTAL:				34.70	124.45		

*** SECOND DEFENDER UNIT PLACED ON ARC NO 3 ***

first path but the presence of several narrow arcs raise the path's cost substantially.

This section has demonstrated the viability of the three algorithms developed for avenue of approach generation. The paths generated contained the desired features of flow rate, best road surface, and minimum travel time. The results validate the concepts of tailoring avenue of approach generation to variable conditions such as weather and enemy objective, composition, and doctrine. The potential for the development of further algorithms is evident.

B. ARC COMBAT

This section will demonstrate the combat process as it occurs on several types of arcs. Of interest is the impact of the arc flow rate for attacker deployment on the duration and outcome of the battle. In each example, the defender has a value of 500 SUA and the attacker's value is 1500. Breakpoints are 50% and 30% strength for attacker and defender, respectively. The Helmboldt equation Weiss parameter is .5 for each example with the result that the combat process approximates Lanchester's Linear Law. Time-steps remain constant at two minutes each. In addition to the effect of arc flow rate on the battle, the impact of artillery fires will be demonstrated in the third example.

1. Wide Arc Battle Results

The first example is the arc battle described in Figure 5.1 . Arc characteristics and individual time-step attrition is provided in Appendix C. The One Km width of this arc results in a relatively short battle as the attacker can quickly deploy across the wide off-road area. Battle duration is composed of 8 steps of deployment and 4 steps of movement for the attacker. Attacker force level increases quickly as a result of the high flow rate of the arc and is indicated by the ATKR ONARC line in Figure 5.1 .

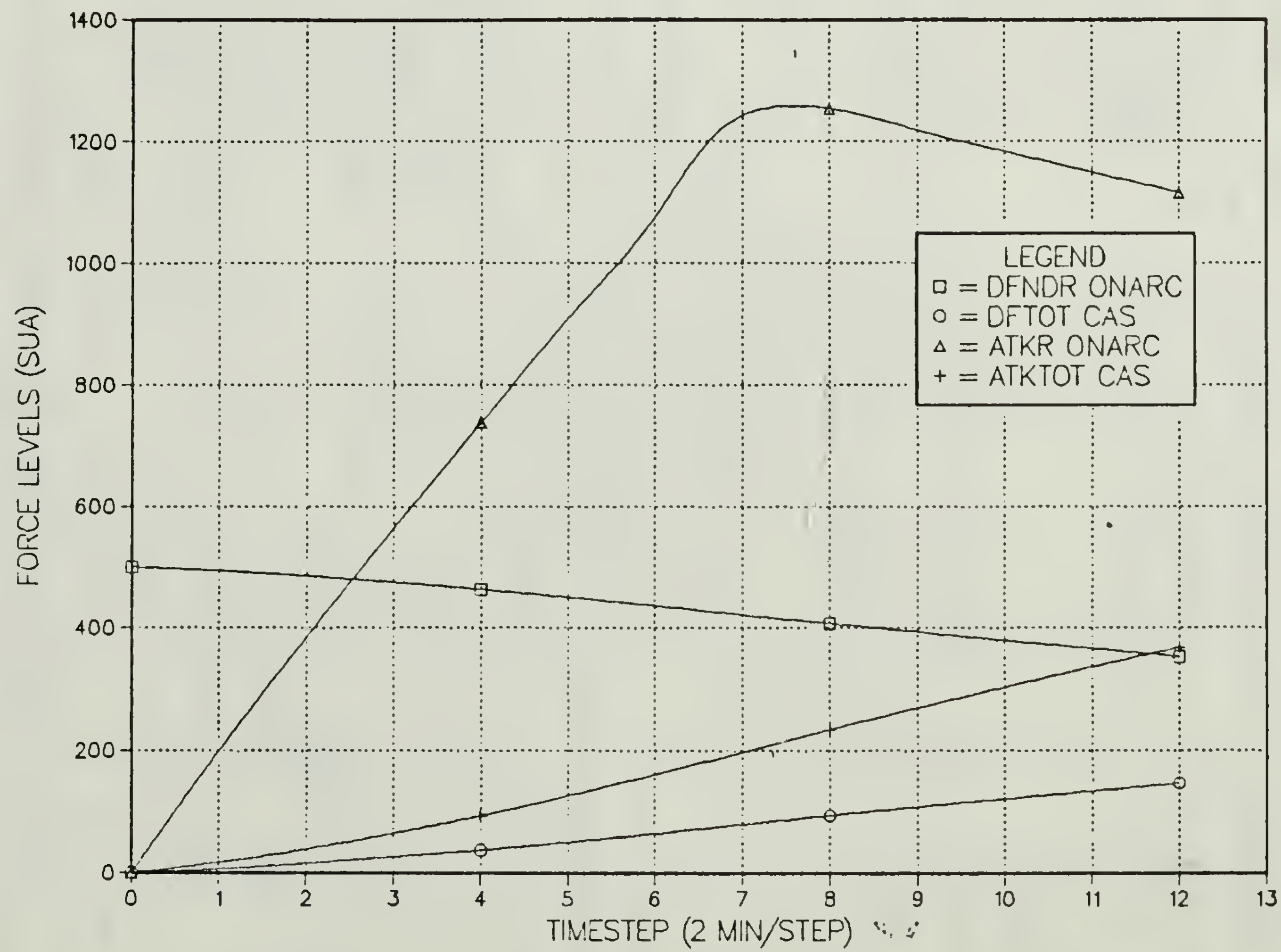


Figure 5.1 Wide Arc Battle Results.

The rise in attacker force level increases attacker and defender casualties in accordance with the Linear Law and is reflected in the increasing values of the DFTOT CAS and ATKTOT CAS lines in the figure. The defender force level, DFNDR ONARC, steadily declines as a result of attacker attrition and not being reinforced. The battle terminates at step 12 when battle duration elapses. The peak of the ATKR ONARC line at step 7 indicates the completion of the attacker's deployment on the arc.

The results bear out the defender's aversion for this type of arc. The large off-road width results in a faster attacker deployment rate, shorter battle duration and delay times, and higher defender attrition.

2. Narrow Arc Battle Results

The second example is a battle fought on a comparatively narrow 110 meter wide arc and are displayed in Figure 5.2 . Arc characteristics and individual time-step attrition is provided at Appendix D. Due to the limited manuever space, it takes the attacker a much greater time to deploy his forces and his build up is much slower. These conditions favor the defender. The battle terminates when the attacker reaches breakpoint at time-step 54. Casualty percentages are 51 and 61 for the attacker and defender, respectively. As discussed earlier, the delay time value is set to a high number, 999, to show the great value of this arc to the defender. The principle feature of this arc is the longer battle duration imposed by the reduced arc flow rate. The attacker deploys onto the arc much more slowly than on the wide arc. Attrition ratios do not vary from the wide arc due to the value of the Weiss parameter approximately modelling the Linear Law. The loss rate is slower than the previous example but the longer duration of the battle permits the battle to continue long enough for the attacker to break. This arc is more desirable for the

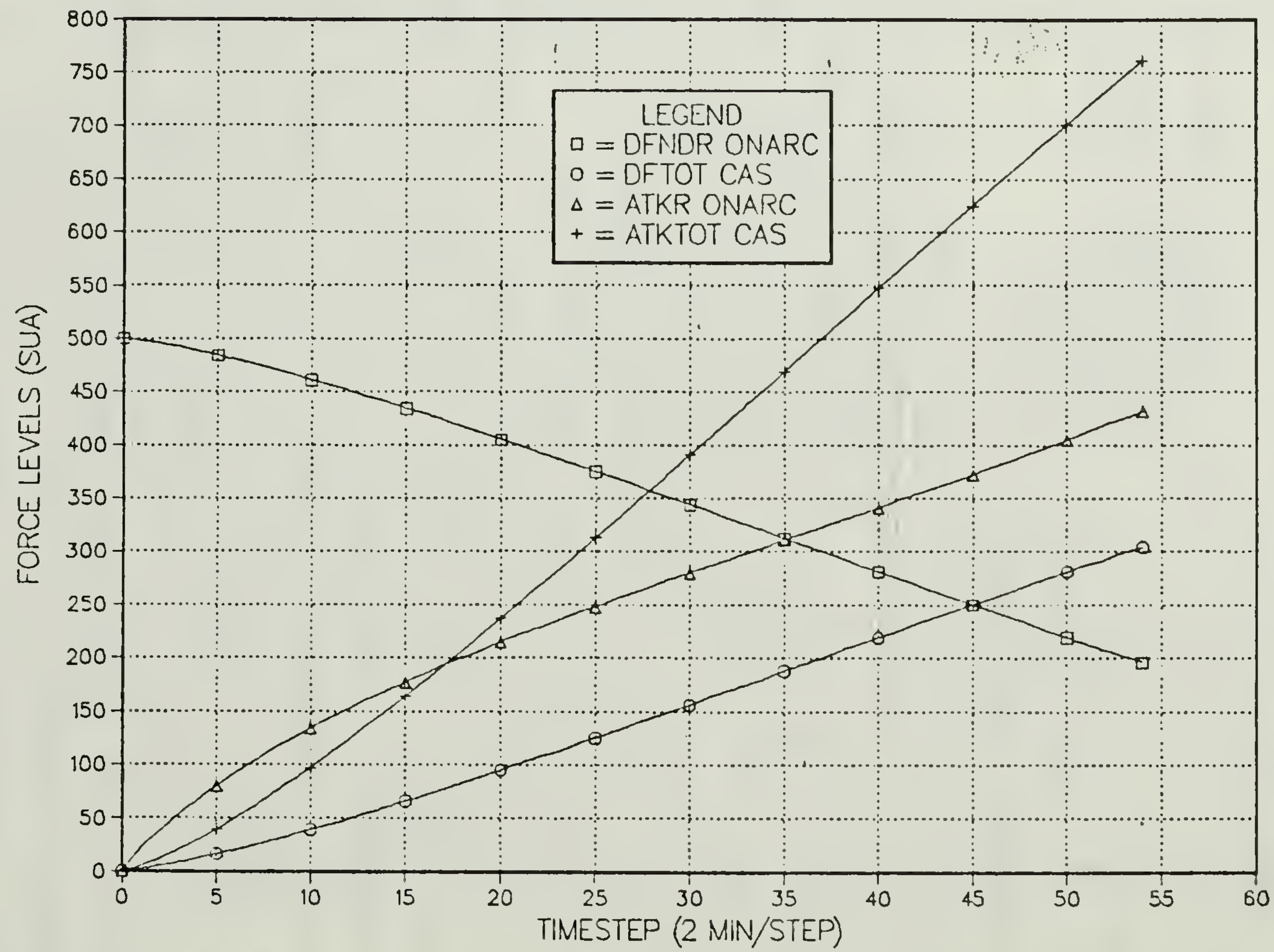


Figure 5.2 Narrow Arc Battle Results.

defense than the previous example. The effects of the restrictive terrain and resultant diminished flow rate were captured in the simulation and their benefit to the defender demonstrated.

3. Narrow Arc Battle With Indirect Fire

The last example of this section will demonstrate the results of incorporating indirect fire in the narrow arc battle. The effects of indirect fire are shown in Figure 5.3 . Arc characteristics and inividual time-step attrition are provided at Appendix E. Each side is assumed to have a six-tube battery available. This is reasonable for a defender with a priority of fires and an attacker whose support is leap-frogging to stay up with him. Multiple batteries, however, can be easily included in the model. Equal rates of fire of three rounds per minute are given and the tube allocations are 30 and 90 for defender and attacker, respectively. Both sides are assumed to begin indirect fire at the start of the direct fire battle, as discussed in Chapter IV. This is a disadvantage for the defender despite the greater effectiveness of his fire. The defender's limited number of rounds land when the enemy has comparatively few people at risk. The attacker benefits from his greater ammunition supply. The slope changes at t = 6 and t = 10 for the attacker and defender respectively are causd by the cessation of the opponent's indirect fire and resultant lower attrition. The attacker's advantage in artillery is reflected in the battle outcome: the defender breaks at time-step 41.

Note that this arc has the same features as the arc in the previous example. What was a good arc for the defender has now become infeasible!

C. ARC SELECTION

The last section of this chapter will examine the arc selection process that occurs after each arc in sector on

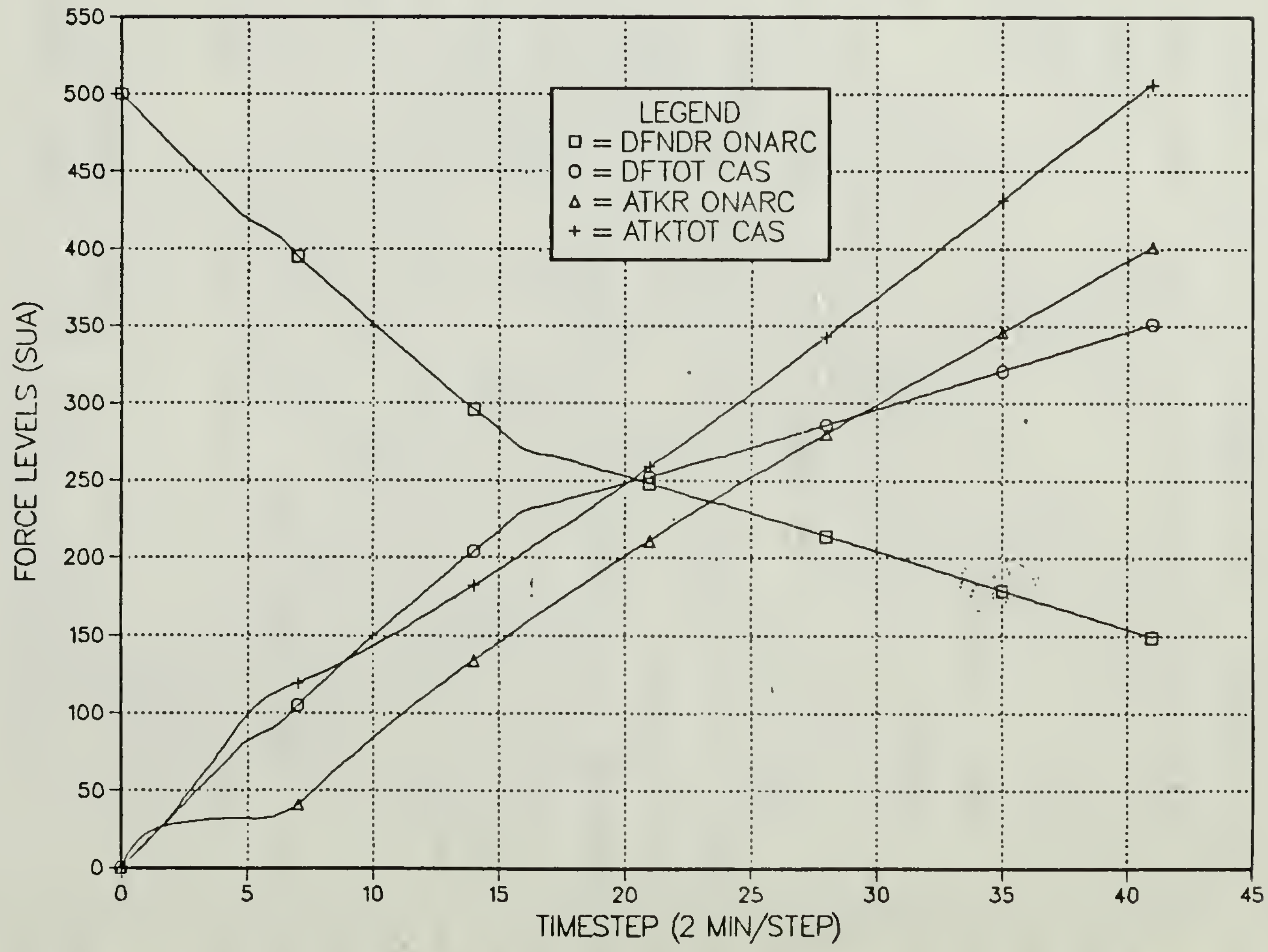


Figure 5.3 Narrow Arc Battle with Indirect Fire.

the avenue of approach has completed its simulated battle. Table V displays the key factors in selecting the arc for unit placement. Column descriptions, where similar, have the same meaning as described in Appendix C. Several new columns have been introduced. The WEIGHT FACTOR represents the weighting function developed in Equation 3.1 . DELAY TIME and OBJ TIME represent the battle delay times before and after multiplication by the weighting function FLOTFN.

The selection process is simply a matter of picking the arc with the highest OBJ TIME. On the present avenue, arcs five and six broke the enemy. Despite being in the rear of the sector and receiving a relatively low weighting function value, these arcs are valuable because they alone stopped the enemy. Not surprisingly, these two arcs were the narrowest and had the lowest flow rate.

The results of the arc evaluations provide other useful information besides initial unit and obstacle placement. The next few high OBJ TIME arcs to the rear of the selected arc could be identified as supplementary positions. If most of the initial unit locations fall in the rear of the sector, adjustments to the sector boundary or the unit's defensive plan might be considered. A delay back to the optimal positions in the rear of the sector may be appropriate or the unit may request an extension of its rear boundary to provide more depth to the sector.

TABLE V

ARC SELECTION PROCESS

ARC NO	ARC TYPE	NO. LNS	OFF-RT WD (KM)	OFF-RT CLASS	LENGTH (KM)	COST	FLOW RT (SUA/MIN)	WEIGHT FACTOR	DELAY TIME	OBJ TIME	POINTER
1	DIRT. ROAD	2	0. 30	1	3. 00	0. 15	53. 82	2. 00	180. 55	361. 09	1322704
2	ASPHALT. RD	2	0. 30	2	1. 50	0. 08	53. 82	1. 80	163. 42	293. 47	1323728
3	DIRT ROAD	2	0 50	2	3. 20	0 16	88. 54	1 59	142. 76	227. 22	1331728
4	CONCRETE RD	2	0. 50	2	1. 40	0. 07	88. 54	1. 37	123. 45	168. 70	1342160
5	CONCRETE RD	2	0 10	1	1. 00	0. 05	19. 10	1. 26	999. 00	1260. 52	1344208
6	CONCRETE RD	5	0 05	1	1. 00	0. 05	13. 02	1 16	999. 00	1161. 14	1349136
7	CONCRETE. RD	2	1. 00	1	2. 00	0 10	175. 35	1. 00	87. 31	87 31	1349840

BEST ARC DELAY = 1260. 52 AT ARC 1344208

XXXXXX NO UNITS ON ADJACENT ARCS, GO TO PLACE. UNIT XXXX

XXXXXXX RESULTS XXXXXXX

ADDITIONAL 500. 00 SUA ADDED TO ARC 1344208
TOTAL SUA ON ARC = 500. 00
OBSTACLE TYPE ADDED = BLK2DRY2
OBSTACLE DELAY TIME ADDED = 45. 00 MINUTES
TOTAL OBSTACLE DELAY TIME THIS ARC = 45. 00 MINUTES
TOTAL BATTLE DELAY TIME THIS ARC = 1260 52 MINUTES

VI. ANALYSIS OF HELMBOLDT EQUATION IN DIRECT FIRE ATTRITION

A. NARROW ARC DIRECT FIRE ATTRITION

This chapter will analyze the results of varying the Weiss parameter of the Helmboldt equation used in determining direct fire attrition. Four values of the Weiss parameter, 0, .5, .70, and 1 will be tested in combat simulations on a narrow (.11 Km) and a wide (1.01 Km) arc.

It should be noted that the Weiss parameter in the range (0., 0.5) represents non-combat losses in that the losses to force x in Equation 6.2 are driven by the number of units surviving. For the range (0.5, 1.), the losses to x range from a Linear to a Square Law Lanchesterian process. The value of Weiss parameter equal zero is included only for purposes of comparison.

The results of the narrow arc direct fire attrition process are displayed in Figures 6.1 and 6.2 . In this situation, the attacker is defeated under all values of the Weiss parameter. As shown in Figure 6.1, the attacker reaches breakpoint quickest when the Weiss parameter equals one. In this case attrition is determined by Lanchester's Square Law. (see Taylor [Ref. 6: p. 39] for this and subsequent references to Lanchesterian forms.) Under these conditions, Equation 4.1 reduces to:

$$dx/dt = -a * y \qquad \text{(eqn 6.1)}$$

where:

dx/dt = Attacker attrition over time (SUA/Min).

-a = The defender's attrition production coefficient (Percentage/Min).

y = Value of the defender currently on the arc.

The attacker, unable to build up forces quickly on the narrow terrain, inflicts minimal casualties on the defender over the entire battle. The defender, having used the terrain to full advantage, has his total value to fight with. He destroys the attacker forces as fast as they deploy. While this process may be accurate during the initial stages of the battle if the defender has surprise, it is not likely to be the case during later stages as the defender becomes tired and runs low on ammunition while the attacker commits fresh troops, gradually discerns the defender's locations, and employs tactics that reduce his vulnerability. Zero, then, does not appear to be a good value for the Weiss parameter in this attrition process.

Attacker losses develop slowest, and the defender's losses are greatest, when the Weiss parameter equals zero. This value transforms Equation 4.1 into the Logarithmic Law for which:

$$dx/dt = -a * x \qquad \text{(eqn 6.2)}$$

where:

x = Value of the attacker currently on the arc and other terms retain their previous definitions.

Unit attrition is proportional to the number, or value, of the units present. Losses produced with this equation are normally attributed to non-combat actvities. The defender, at high values initially, experiences his greatest attrition under this parameter value. The attacker's attrition starts out slowly, but increases as the value of his forces on the arc increase. A Weiss parameter value of zero does not appear to be the best in the current scenario. The defender, having selected and prepared the battlefield, should have his attrition influenced by the level of attackers present and not just his own force levels.

Midway between these first two examples is the Weiss parameter value of .5 . This value transforms Equation 4.1 into an approximation of Lanchester's Linear Law:

$$dx/dt = -a * x * y \qquad \text{(eqn 6.3)}$$

where y is the value of the attacker present on the arc and other terms retain their previous definitions.

Determining attrition with the product of the two sides' force levels favors the attacker initially as it reduces the effect of the defender's concentration on the narrow arc. At W = .7, casualty figures lay between the Linear Law representation and the Square Law representation. At this level, the attrition process leans toward the Linear Law but the force ratios are still important determinants to the battle outcome.

B. WIDE ARC DIRECT FIRE ATTRITION

The results of the wide arc battle are displayed in Figures 6.3 and 6.4 . The impact of the attacker's greatly increased ability to deploy on this type of arc is evident in the attrition curves. The defender now suffers his greatest attrition under the Square Law while the attacker suffers his least. The attacker's greatest attrition and the defender's least losses, come under the Logarithmic Law. Here, just the present value of the force determines its non-combat loss rate. The Linear Law results again lie between these two extremes. With a Weiss value of .7, the attacker benefits from improved force ratio on terrain more favorable to him with this figure, while the battle process leans toward the Linear Law.

The Weiss parameter might also be used as an arc attibute. It would serve as a surrogate for the effects of factors on the attrition process such as cover, concealment,deployment area, etc. The use of the Weiss parameter in this role is an area for future research.

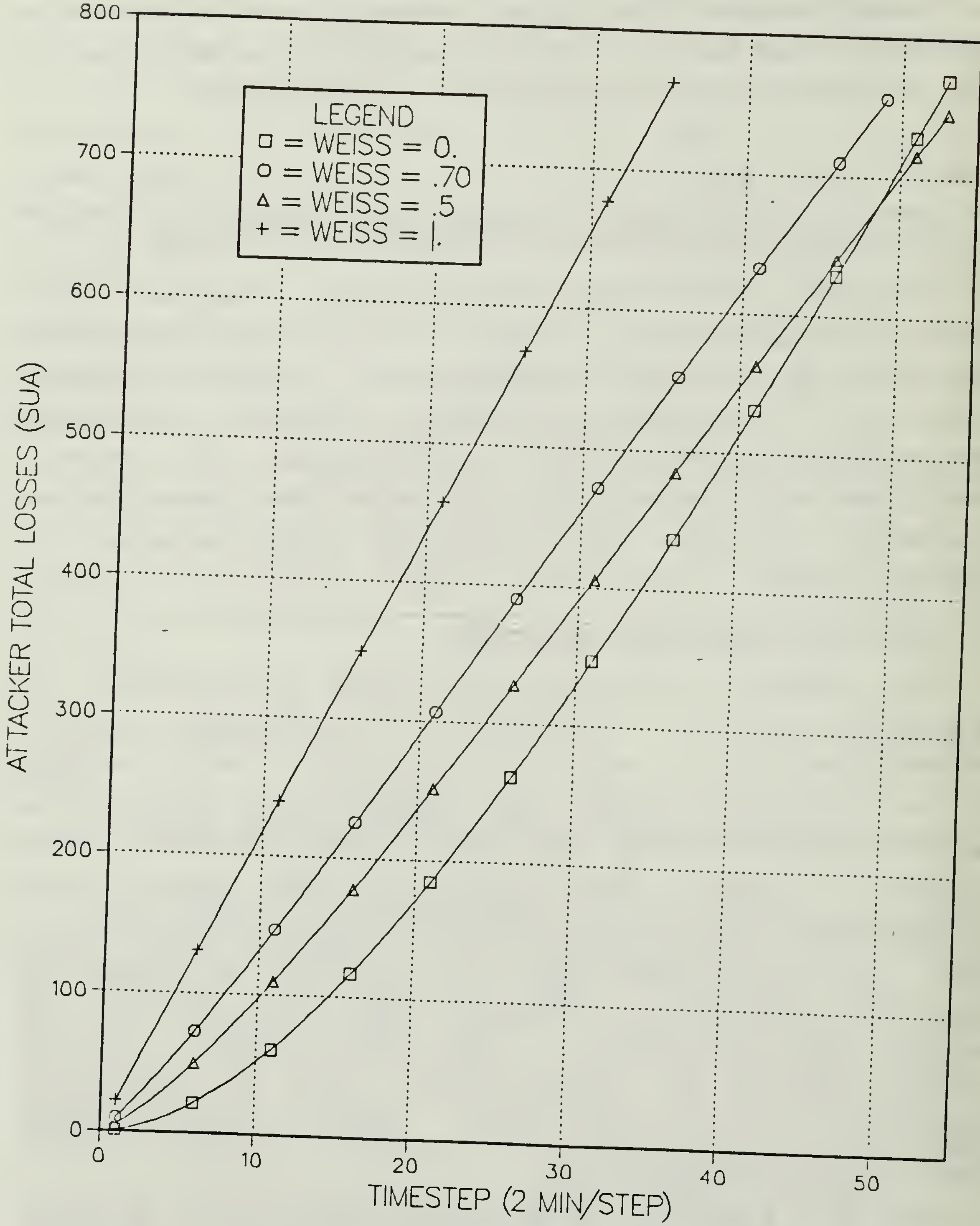


Figure 6.1 Attacker Sensitivity to Varying Weiss Parameter, Narrow Arc.

This chapter has explored the sensitivity of the direct fire attrition process to variation in the Weiss parameter of the Helmboldt attrition equation. In the case of the narrow arc battle, the attrition process seems to be relatively unaffected by variation in the Weiss parameter, the exception being the Square Law case. The impact of the terrain and the resultant inability of the attacker to bring many direct fire units into battle overshadows parameter values.

The wide arc battle shows large sensitivity to parameter changes. The greater deployment room for the attacker allows him to improve his force ratios over the conditions he faced on the narrow arc. As a result, the parameter values, which effect the significance of these force ratios, now have a larger impact on attrition outcomes.

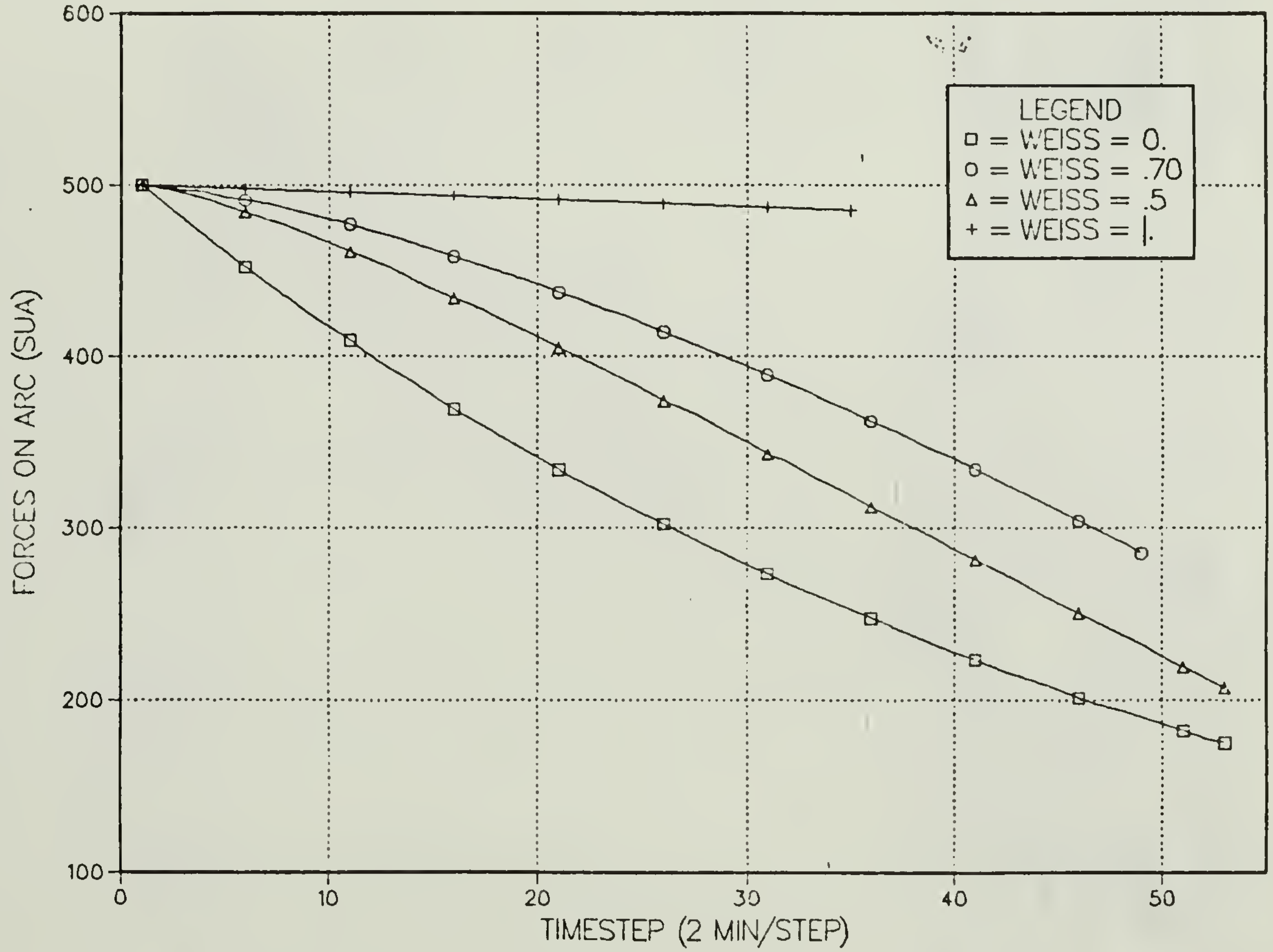


Figure 6.2 Defender Sensitivity to Varying Weiss Parameter, Narrow Arc.

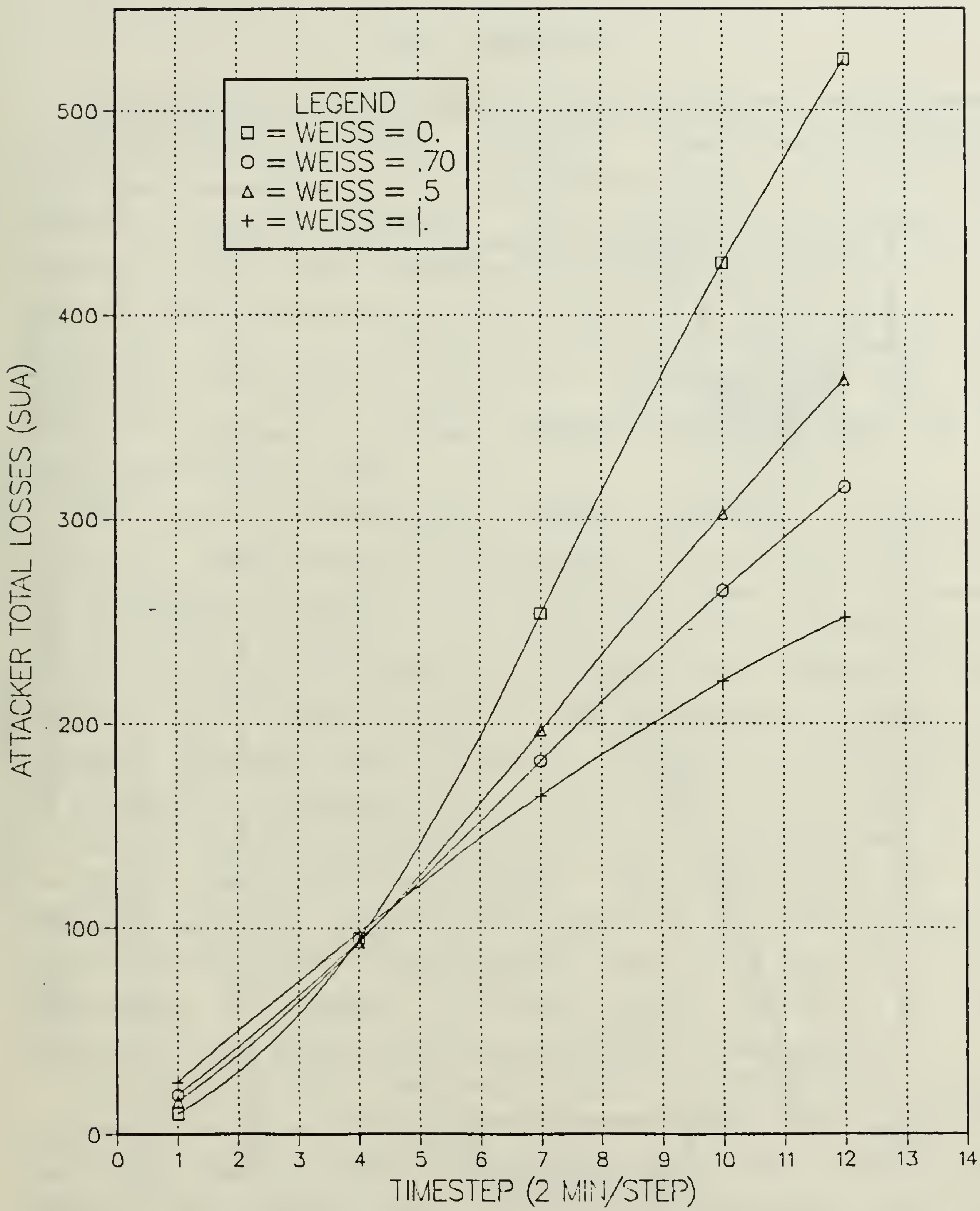


Figure 6.3 Attacker Sensitivity to Varying Weiss Parameter, Wide Arc.

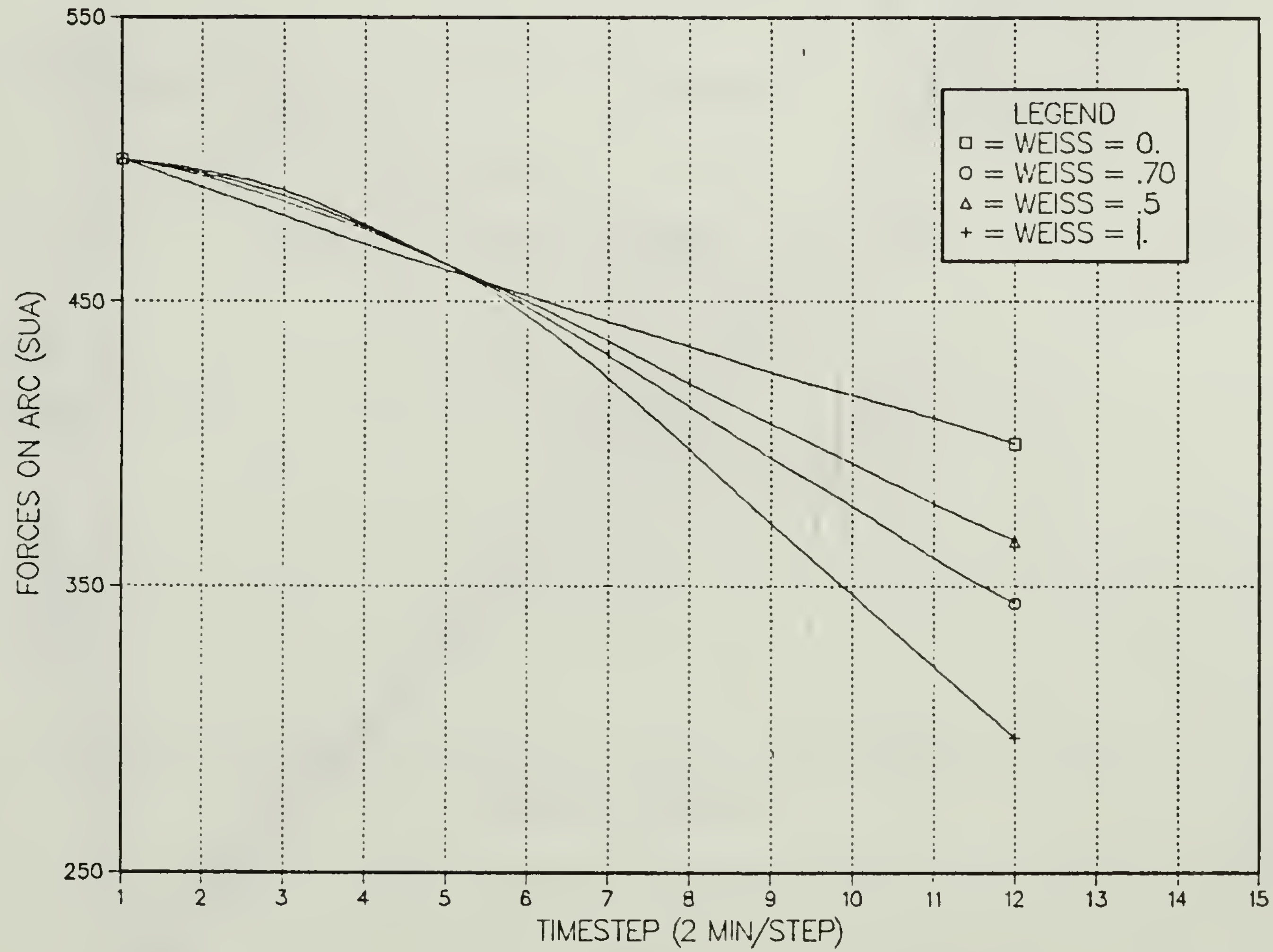


Figure 6.4 Defender Sensitivity to Varying Weiss Parameter, Wide Arc.

VII. CONCLUSION

This thesis has achieved the objectives which were established in the initial chapter. Shortcomings noted in previous research on unit and engineer asset distribution in ALARM have been redressed. The problem Craig noted of several units being unnecessarily placed around a single node has been corrected by an algorithm which moves a unit to a node to control more than one arc. This permits the modelling of a unit's ability to influence more than one avenue of approach, either directly or through the use of supplementary positions.

The lack of integration between placement of manuever units and countermobility operations has been corrected by developing an algorithm which combines these two processes and allows the potential of the terrain for countermobility operations to enter into the arc evaluation process. The methodology is very flexible and accomodates changes in engineer resources and the type and method of construction of obstacles. Obstacles will now be located on the same arc as the unit and their coverage by fire is thus ensured.

This thesis has reaffirmed the usefulness of the Humboldt equation for research into the attrition process. It is attractive in the present case as a result of the importance it places on force ratios as well as its flexibility to model the classic forms of Lanchester.

The desirability and feasibility of alternative algorithms for avenue of approach generation and arc selection have been demonstrated. Depending upon force doctrine and METT-T, the optimal forms of these two parameters can vary widely. Either a mechanism to build these algorithms within the model is needed, or algorithms should be developed which would be selected based upon an evaluation of the factors mentioned above.

Three criteria for avenue of approach generation were explored. An algorithm which produced the fastest travel time avenue was developed. Appropriate for reconnaissance or raiding elements, the MIN_TIME path linked many different types of arcs together to form the fastest, and normally the shortest, route. The BEST_ROAD path found the avenue with the best and widest road surfaces. This path produced routes with excellent features for an enemy with heavy logistic requirements. Like the MIN_TIME route, this path frequently contained chokepoints for easy defense. The third algorithm, BEST_FLOW, developed the path with the best off-route manueverability. An attacker equipped with mobile forces and enjoying numerical superiority would find this path attractive, at least for the assaulting echelons. Wide arcs would allow him to take full advantage of his higher level of forces in the event of an arc battle.

Two methods of selecting the appropriate arc on the avenue were developed. In one instance, the arc that produced the greatest delay upon the attacker, as a result of casualties and battle duration, was selected. This can often select arcs far to the sector rear. Alternatively, an arc can be evaluated by considering both its time delay production and its proximity to the FLOT. Between two arcs of equal delay potential, the one closest to the FLOT would be selected. A linear weighting function was used in this thesis to represent the value of a FLOT position against a rear position, but this can be easily changed to other types of functions. For example, a defense along a river bank might require an exponential function to represent the importance of defending along the shoreline.

In both cases, a combat simulation was conducted to generate casualties and assess the delay produced by an arc. Central to this simulation was the concept of flow rate, the ability of the terrain to physically support deployed

forces. This process, using a few arc attributes and doctrinally derived numbers, allows representation and analysis of terrain more cheaply than other forms of terrain representation and captures the constraints terrain places on a deploying force. As with avenue generation, wide flexibility must exist in the development of arc selection algorithms. Doctrine and METT-T will determine the type of battle the defender wants to fight and the kind of terrain he wants to use.

This thesis illuminates areas for further research. The development of additional defense algorithms, as well as offense planning algorithms, is needed. The whole area of plan verification must be developed. Having placed units and obstacles on the network, an attacker must then be introduced to the sector and combat simulations run to determine if the plan satisfies mission requirements. If requirements are not met, appropriate procedures must be available that allocate additional resources or query higher echelon for assistance. Algorithms for the movement of units to alternate and suplementary positions are needed. The engineer SOP Table can be expanded to accomodate new counter-mobility resources and methods to model the effects of position preparation should be developed. In sum, the area of unit and engineer asset allocation and control is one with challenges for ALARM's continued dedvelopment.

APPENDIX A

ARC AND NODE CHARACTERISTICS

1. ARC TYPES:

[Ref. 1: p. 43]

Type	Code
Autobahn	1
Autobahn w/ Railroad	2
Railroad	3
Concrete Road	4
Asphalt Road	5
Pint Road	6
Forest	7
Open Country	8
Road and Railroad	9
Bridge, Tunnel	10

2. ARC ATTRIBUTES:

Identification number of arc

Identification number for end nodes

Number of lanes on route

Off-route classification

Main route classification

Battle delay time of arc, given some allocated defender and/or obstacles.

Basic speed possible for attacker

Width of arc for deployment of type units allowed by off-route classification

Acquisition range (subjective aggregation for the entire arc).

Value of all defendeing forces on the arc.

Obstacle delay time created by all defender countermobility operations on the arc.

Additionally, each arc owns a set of obstacles and a set of defending units that are currently assigned to the arc.

3. NODE TYPES:

Type	Code
City	1
Village	2
Autobahn Junction	3
Road Junction	4
Hill Top	5
Other	6

4. NODE ATTRIBUTES:

Identification Number

Latitudinal and Longitudinal Coordinates

Node type

APPENDIX B

ARC ROUTE CLASSIFICATION

[Ref. 1: p. 44]

Code	Type Unit Terain will Support
1	Heavy Tank
2	Medium Tank & Fighting Vehicle
3	Heavy Truck
4	Light Truck
5	Dismounted Toops

APPENDIX C
WIDE ARC COMBAT PROCESS

This Appendix contains the detailed results of the combat on a wide arc. Table VI is the reference for the following discussion. The attacker is attempting to move down an arc 1.01 Km wide and 2 Km long, and no obstacles currently exist on the arc. The best feasible obstacle available for this arc is Block Secondary Road, type 2. Terrain and potential obstacle limitations produce a total flow rate of 100.2 SUA/Min for the arc. The number of timesteps required to move the attacker to the defender's position, as described in Chapter III are 8 and 4, respectively.

The next group of numbers in the table describe the attrition coefficients used in the arc battle. These variables correspond to those developed in the attrition equations of Chapter IV.

Due to the relatively large width of the arc, the time required to deploy the attacker on the arc is small and the attacker is able to quickly build up his forces engaged in battle. The increasing value of the ATTKR ONARC column shows the effect of the deployment of the attacker on the arc through step 8. The defender, not being reinforced, steadily declines in strength in the DFNDR ONARC column. The ATKR DFCS and DFDR DFCS columns indicate the corresponding casualties from direct fire during the timesteps. The next four columns describe attacker and defender casualties from attack helicopters and artillery and have no value because these modules were not included for this demonstration. The last columns simply recapture the total casualties (SUA) each side has lost through the end of the current

TABLE VI

WIDE ARC BATTLE DETAILED RESULTS

TYPE ARC	CONCRETE. RD
HAS DEPLOYABLE WIDTH OF	1.01 KM
LENGTH OF	2.00 KM
POINTER =	1349840
DEPLOYABLE VELOCITY IS	10.00 KM/HR
UNRESTRICTED FLOW RATE FOR THIS ARC IS	175 35 SUA/MIN
FOR THIS ARC BEST OBST DELAY IS	45.00 MINUTES
WITH OBSTACLE TYPE =	BLK2DRY2
PREVIOUS OBSTACLE DELAY ON ARC =	0. MINUTES
MAX FLOW RATE WITH PRIOR & 1 NEW OBST IS	100.20 SUA/MIN
NO. OF STEPS TO DEPLOY ATTACKER ON ARC IS	8.0
NO. OF STEPS TO MOVE FORCE DOWN ARC IS	4.0
NO. SUA CURRENTLY ON ARC =	0.
NO. SUA TO PLACE THIS CYCLE =	500.00

DFNDR ATRT COEF	ATKR ATRT COEF	DEF WEISS	ATKR WEISS	AIFCAS	DIFCAS	AHL CSRT	DH CSRT
0.010	0.025	0.500	0.500	0.01	0.20	3.00	5.00

DEF BREAKPOINT IS 150.0 ATKR BREAKPOINT IS 750.0

STP	ATKR DPLYD	ATTKR ONARC	DFNDR ONARC	ATKR DFCS	DFDR DFCS	ATTKR HLCAS	DFDR HLCS	ATTKR ARTCS-	DFDR ARTC	ATKTD CAS	DFTOT CAS
1	200.4	200.4	500.0	15.8	6 3	0.	0.	0.	0.	15.8	6.3
2	400.8	385.0	493.7	21.8	8.7	0.	0.	0.	0.	37.6	15.0
3	601.2	563.6	485.0	26.1	10.5	0.	0.	0.	0.	63.8	25 5
4	801.6	737.8	474.5	29 6	11.8	0.	0.	0.	0.	93.3	37.3
5	1002.0	908.6	462.7	32.4	13.0	0.	0.	0.	0.	125.8	50.3
6	1202.4	1076.6	449.7	34.8	13.9	0.	0.	0.	0.	160.6	64 2
7	1402.8	1242.2	435.8	36.8	14.7	0.	0.	0.	0.	197.3	78.9
8	1451.4	1254 0	421.1	36.3	14.5	0.	0.	0.	0.	233.7	93.5
9	1451.4	1217.7	406.5	35 2	14 1	0.	0.	0.	0.	268.9	107 5
10	1451.4	1182.5	392.5	34 1	13.6	0.	0.	0.	0.	302.9	121.2
11	1451.4	1148.5	378.8	33.0	13.2	0.	0.	0.	0.	335.9	134.4
12	1451.4	1115.5	365.6	31.9	12.8	0.	0.	0.	0.	367.8	147.1

XXXX NEITHER SIDE BREAKS NEW ARC TIME IS 87.31 MIN XXXX

time-step. At the bottom of the figure, the results of the battle are summarized and the total delay time displayed. These conditions preclude the defender from breaking the attacker. The attacker does not have great difficulty in moving down the arc and forcing the defender to withdraw. Casualties are 25 and 29 percent for the attacker and defender, respectively. The delay time created by this battle on the attacking battalion, at 2.5 minutes per percent attacker casualty plus the battle duration is 87 minutes. Judged by the simulation results, this arc does not appear to be attractive for unit placement because it failed to break the attacker, generated relatively little delay time, and had higher casualty ratios for the defender.

APPENDIX D

NARROW ARC COMBAT PROCESS

This appendix contains the detailed results of the narrow arc battle. Variables and columns retain the definitions provided in Appendix C. The small off-road area available to the attacker for deployment provides a slower buildup of attacker force level, a larger battle duration time and allows the defender to break the attacker at time-step 54.

TABLE VII

NARROW ARC BATTLE DETAILED RESULTS

TYPE ARC	CONCRETE RD
HAS DEPLOYABLE WIDTH OF	0.11 KM
LENGTH OF	1.00 KM
POINTER =	1344208
DEPLOYABLE VELOCITY IS	10.00 KM/HR
UNRESTRICTED FLOW RATE FOR THIS ARC IS	19.10 SUA/MIN
FOR THIS ARC BEST OBST DELAY IS	45.00 MINUTES
WITH OBSTACLE TYPE =	BLK2DRY2
PREVIOUS OBSTACLE DELAY ON ARC =	0. MINUTES
MAX FLOW RATE WITH PRIOR % 1 NEW OBST IS	10.91 SUA/MIN
NO. OF STEPS TO DEPLOY ATTACKER ON ARC IS	70.
NO. OF STEPS TO MOVE FORCE DOWN ARC IS	3.
NO. SUA CURRENTLY ON ARC =	0.
NO. SUA TO PLACE THIS CYCLE =	500.00

DFNDR ATRT COEF	ATKR ATRT COEF	DEF WEISS	ATKR WEISS	AIFCAS	DIFCAS	AHL CSRT	DH CSRT
0.010	0.025	0.500	0.500	0.01	0.20	3.00	5.00

DEF BREAKPOINT IS 150.0 ATKR BREAKPOINT IS 750.0

STP	ATKR DPLYD	ATTKR ONARC	DFNDR ONARC	ATKR DFCS	DFDR DFCS	ATTKR HLCAS	DFDR HLCS	ATTKR ARTCS	DFDR ARTC	ATKTO CAS	DFTOT CAS
1	21.8	21.8	500.0	5.2	2.1	0.	0.	0.	0.	5.2	2.1
2	43.7	38.4	497.9	6.9	2.8	0.	0.	0.	0.	12.1	4.9
3	65.5	53.3	495.1	8.1	3.3	0.	0.	0.	0.	20.3	8.1
4	87.3	67.0	491.9	9.1	3.6	0.	0.	0.	0.	29.3	11.7
5	109.1	79.8	488.3	9.9	3.9	0.	0.	0.	0.	39.2	15.7
6	131.0	91.7	484.3	10.5	4.2	0.	0.	0.	0.	49.8	19.9
7	152.8	103.0	480.1	11.1	4.4	0.	0.	0.	0.	60.9	24.3
8	174.6	113.7	475.7	11.6	4.7	0.	0.	0.	0.	72.5	29.0
9	196.4	123.9	471.0	12.1	4.8	0.	0.	0.	0.	84.6	33.8
10	218.3	133.7	466.2	12.5	5.0	0.	0.	0.	0.	97.1	38.8

TABLE VII(Cont'd)

STP	ATKR DPLYD	ATTKR ONARC	DFNDR ONARC	ATKR DFCS	DFDR DFCS	ATTKR HLCAS	DFDR HLCS	ATTKR ARTCS	DFDR ARTC	ATKTO CAS	DFTOT CAS
11	240.1	143.0	461.2	12.8	5.1	0.	0.	0.	0.	109.9	44.0
12	261.9	152.0	456.0	13.2	5.3	0.	0.	0.	0.	123.1	49.2
13	283.7	160.7	450.8	13.5	5.4	0.	0.	0.	0.	136.5	54.6
14	305.6	169.0	445.4	13.7	5.5	0.	0.	0.	0.	150.2	60.1
15	327.4	177.1	439.9	14.0	5.6	0.	0.	0.	0.	164.2	65.7
16	349.2	185.0	434.3	14.2	5.7	0.	0.	0.	0	178.4	71.3
17	371.0	192.7	428.7	14.4	5.7	0.	0.	0.	0.	192.7	77.1
18	392.9	200.1	422.9	14.5	5.8	0.	0.	0.	0.	207.3	82.9
19	414.7	207.4	417.1	14.7	5.9	0.	0.	0.	0.	222.0	88.8
20	436.5	214.5	411.2	14.9	5.9	0.	0.	0.	0.	236.8	94.7
21	458.3	221.5	405.3	15.0	6.0	0.	0.	0.	0.	251.8	100.7
22	480.2	228.3	399.3	15.1	6.0	0.	0.	0.	0.	266.9	106.8
23	502.0	235.1	393.2	15.2	6.1	0.	0.	0.	0.	282.1	112.8
24	523.8	241.7	387.2	15.3	6.1	0.	0.	0.	0.	297.4	119.0
25	545.6	248.2	381.0	15.4	6.2	0.	0.	0.	0.	312.8	125.1
26	567.5	254.7	374.9	15.4	6.2	0.	0.	0.	0.	328.2	131.3
27	589.3	261.0	368.7	15.5	6.2	0.	0.	0.	0.	343.8	137.5
28	611.1	267.4	362.5	15.6	6.2	0.	0.	0.	0.	359.3	143.7
29	632.9	273.6	356.3	15.6	6.2	0.	0.	0.	0.	374.9	150.0
30	654.8	279.8	350.0	15.6	6.3	0.	0.	0.	0.	390.6	156.2
31	676.6	286.0	343.8	15.7	6.3	0.	0.	0.	0.	406.3	162.5
32	698.4	292.2	337.5	15.7	6.3	0.	0.	0.	0.	422.0	168.8
33	720.2	298.3	331.2	15.7	6.3	0.	0.	0.	0.	437.7	175.1
34	742.1	304.4	324.9	15.7	6.3	0.	0.	0.	0.	453.4	181.4
35	763.9	310.5	318.6	15.7	6.3	0.	0.	0.	0.	469.1	187.7
36	785.7	316.6	312.3	15.7	6.3	0.	0.	0.	0.	484.8	193.9
37	807.5	322.7	306.1	15.7	6.3	0.	0.	0.	0.	500.6	200.2
38	829.4	328.8	299.8	15.7	6.3	0.	0.	0.	0.	516.3	206.5
39	851.2	334.9	293.5	15.7	6.3	0.	0.	0.	0.	531.9	212.8
40	873.0	341.1	287.2	15.6	6.3	0.	0.	0.	0.	547.6	219.0
41	894.8	347.3	281.0	15.6	6.2	0.	0.	0.	0.	563.2	225.3
42	916.7	353.5	274.7	15.6	6.2	0.	0	0.	0.	578.8	231.5
43	938.5	359.7	268.5	15.5	6.2	0.	0	0.	0.	594.3	237.7
44	960.3	366.0	262.3	15.5	6.2	0.	0.	0.	0.	609.8	243.9
45	982.1	372.3	256.1	15.4	6.2	0.	0.	0.	0	625.3	250.1
46	1004.0	378.7	249.9	15.4	6.2	0.	0.	0.	0.	640.6	256.3
47	1025.8	385.2	243.7	15.3	6.1	0.	0.	0.	0.	656.0	262.4
48	1047.6	391.7	237.6	15.3	6.1	0.	0.	0.	0.	671.2	268.5
49	1069.4	398.2	231.5	15.2	6.1	0.	0.	0.	0.	686.4	274.6
50	1091.3	404.9	225.4	15.1	6.0	0.	0.	0.	0.	701.5	280.6
51	1113.1	411.6	219.4	15.0	6.0	0.	0.	0.	0.	716.5	286.6
52	1134.9	418.4	213.4	14.9	6.0	0.	0.	0.	0.	731.5	292.6
53	1156.7	425.3	207.4	14.9	5.9	0.	0.	0.	0	746.3	298.5
54	1178.6	432.3	201.5	14.8	5.9	0.	0.	0.	0.	761.1	304.4

XXXX ATTACKER BROKE AT TIMESTEP 54 NEW ARC TIME IS 999.00 MIN XXXX

APPENDIX E

NARROW ARC COMBAT WITH INDIRECT FIRE

This appendix contains the detailed results of the narrow arc battle incorporating indirect fire. Variables and columns retain the definitions provided in Appendix C. The small off-road area available to the attacker for deployment provides a slower buildup of attacker force level and a larger battle duration time. Defender and attacker indirect fire is introduced through time-steps 6 and 16, respectively. The effect of the attacker's indirect fire is great enough to allow him to break the defender at time-step 41. This contrasts the attacker defeat on the same arc in Appendix D in the absence of indirect fire support.

TABLE VIII

DETAILED RESULTS OF NARROW ARC BATTLE WITH INDIRECT FIRE

TYPE ARC	CONCRETE. RD
HAS DEPLOYABLE WIDTH OF	0. 11 KM
LENGTH OF	1. 00 KM
POINTER =	1344208
DEPLOYABLE VELOCITY IS	10. 00 KM/HR
UNRESTRICTED FLOW RATE FOR THIS ARC IS	19. 10 SUA/MIN
FOR THIS ARC BEST OBST DELAY IS	45. 00 MINUTES
WITH OBSTACLE TYPE =	BLK2DRY2
PREVIOUS OBSTACLE DELAY ON ARC =	0. MINUTES
MAX FLOW RATE WITH PRIOR & 1 NEW OBST IS	10. 91 SUA/MIN
NO. OF STEPS TO DEPLOY ATTACKER ON ARC IS	70.
NO. OF STEPS TO MOVE FORCE DOWN ARC IS	3.
NO. SUA CURRENTLY ON ARC =	0.
NO. SUA TO PLACE THIS CYCLE =	500. 00

DFNDR ATRT COEF	ATKR ATRT COEF	DEF WEISS	ATKR WEISS	AIFCAS	DIFCAS	AHL CSRT	DH CSRT
0. 010	0. 025	0. 500	0. 500	0. 01	0. 20	3. 00	5. 00

DEF BREAKPOINT IS 150. 0 ATKR BREAKPOINT IS 750. 0

STP	ATTKR ONARC	DFNDR ONARC	ATKR DFCS	DFDR DFCS	ATTKR HLCAS	DFDR HLCS	ATTKR ARTCS	DFDR ARTC	ATKTO CAS	DFTOT CAS
1	21. 8	500. 0	5. 2	2. 1	0.	0.	10. 6	15. 0	15. 8	17. 1
2	27. 8	482. 9	5. 8	2. 3	0.	0.	13. 5	14. 5	35. 1	33. 9
3	30. 4	466. 1	5. 9	2. 4	0.	0.	14. 7	14. 0	55. 8	50. 3
4	31. 5	449. 7	6. 0	2. 4	0.	0.	15. 3	13. 5	77. 0	66. 1
5	32. 1	433. 9	5. 9	2. 4	0.	0.	15. 6	13. 0	98. 5	81. 5

TABLE VIII (Cont'd)

STP	ATTKR ONARC	DFNDR ONARC	ATKR DFCS	DFDR DFCS	ATTKR HLCAS	DFDR HLCS	ATTKR ARTCS	DFDR ARTC	ATKTO CAS	DFTOT CAS
6	32.5	418.5	5.8	2.3	0.	0.	7.9	6.3	112.2	90.1
7	40.6	409.9	6.4	2.6	0.	0.	0.	12.3	118.6	105.0
8	56.0	395.0	7.4	3.0	0.	0.	0.	11.9	126.1	119.8
9	70.4	380.2	8.2	3.3	0.	0.	0.	11.4	134.2	134.5
10	84.0	365.5	8.8	3.5	0.	0.	0.	11.0	143.0	149.0
11	97.1	351.0	9.2	3.7	0.	0.	0.	10.5	152.2	163.2
12	109.7	336.8	9.6	3.8	0.	0.	0.	10.1	161.8	177.1
13	121.9	322.9	9.9	4.0	0.	0.	0.	9.7	171.8	190.8
14	133.8	309.2	10.2	4.1	0.	0.	0.	9.3	181.9	204.1
15	145.4	295.9	10.4	4.1	0.	0.	0.	8.9	192.3	217.2
16	156.9	282.8	10.5	4.2	0.	0.	0.	8.5	202.8	229.9
17	168.2	270.1	10.7	4.3	0.	0.	0.	0.	213.5	234.1
18	179.4	265.9	10.9	4.4	0.	0.	0.	0.	224.4	238.5
19	190.3	261.5	11.2	4.5	0.	0.	0.	0.	235.6	242.9
20	200.9	257.1	11.4	4.5	0.	0.	0.	0.	246.9	247.5
21	211.4	252.5	11.6	4.6	0.	0.	0.	0.	258.5	252.1
22	221.7	247.9	11.7	4.7	0.	0.	0.	0.	270.2	256.8
23	231.8	243.2	11.9	4.7	0.	0.	0.	0.	282.1	261.6
24	241.7	238.4	12.0	4.8	0.	0.	0.	0.	294.1	266.4
25	251.6	233.6	12.1	4.8	0.	0.	0.	0.	306.2	271.2
26	261.3	228.8	12.2	4.9	0.	0.	0.	0.	318.4	276.1
27	270.9	223.9	12.3	4.9	0.	0.	0.	0.	330.7	281.0
28	280.4	219.0	12.4	5.0	0.	0.	0.	0.	343.1	286.0
29	289.8	214.0	12.5	5.0	0.	0.	0.	0.	355.6	291.0
30	299.2	209.0	12.5	5.0	0.	0.	0.	0.	368.1	296.0
31	308.5	204.0	12.5	5.0	0.	0.	0.	0.	380.6	301.0
32	317.8	199.0	12.6	5.0	0.	0.	0.	0.	393.2	306.0
33	327.0	194.0	12.6	5.0	0.	0.	0.	0.	405.8	311.0
34	336.3	189.0	12.6	5.0	0.	0.	0.	0.	418.4	316.1
35	345.5	183.9	12.6	5.0	0.	0.	0.	0.	431.0	321.1
36	354.7	178.9	12.6	5.0	0.	0.	0.	0.	443.6	326.2
37	363.9	173.8	12.6	5.0	0.	0.	0.	0.	456.2	331.2
38	373.2	168.8	12.5	5.0	0.	0.	0.	0.	468.7	336.2
39	382.5	163.8	12.5	5.0	0.	0.	0.	0.	481.2	341.2
40	391.8	158.8	12.5	5.0	0.	0.	0.	0.	493.7	346.2
41	401.1	153.8	12.4	5.0	0.	0.	0.	0.	506.1	351.2

DEFENSE BROKE AT TIMESTEP 41 NEW ARC TIME IS 166.36 MINUTES

APPENDIX E

PROGRAM CODE

This appendix contains the Simscript routines that were developed for the thesis. These routines are a part of the complete ALARM program on hand at the NPS WAR Lab and will not operate by themselves.

```
ROUTINE ENGINEER.ASSET.UPDATE
GIVEN BEST.ARC.OBSTACLE.TYPE
''*****************************************************************
''*                                                               *
''*   THIS ROUTINE IS CALLED BY DEPASSKTR.R WITH THE TYPE OF      *
''*   OBSTACLE PLACED ON AN ARC DURING THE PREVIOUS EVALUATION    *
''*   OF THE AOA.  IT UPDATES THE ENGINEER ASSET BANK-ACCOUNT     *
''*   BY SUBTRACTING FROM IT THE RESOURCES USED FOR THIS OBSTACLE *
''*                                                               *
''*   NAME:  J MCLAUGHLIN                 DATE:  16 JAN 86        *
''*                                                               *
''*****************************************************************
''DEFINE VARIABLES
DEFINE OBSTACLE,BEST.ARC.OBSTACLE.TYPE AS   INTEGER VARIABLES
LET OBSTACLE = BEST.ARC.OBSTACLE.TYPE
IF OD.SQUAD.HOURS(OBSTACLE) GT 0
   SUBTRACT OD.SQUAD.HOURS(OBSTACLE) FROM AVAILABLE.SQUAD.HOURS
ALWAYS
IF OD.DOZER.HOURS(OBSTACLE) GT 0
   SUBTRACT OD.DOZER.HOURS(OBSTACLE) FROM AVAILABLE.DOZER.HOURS
ALWAYS
IF OD.FUEL.GALLONS(OBSTACLE) GT 0
   SUBTRACT OD.FUEL.GALLONS(OBSTACLE) FROM AVAILABLE.FUEL.GALLONS
ALWAYS
IF OD.MOPHS(OBSTACLE) GT 0
   SUBTRACT OD.MOPHS(OBSTACLE) FROM AVAILABLE.MOPHS
ALWAYS
IF OD.BRIDGE.DEMO.KITS(OBSTACLE) GT 0
   SUBTRACT OD.BRIDGE.DEMO.KITS(OBSTACLE) FROM AVAILABLE.BRIDGE.DEMO.KITS
ALWAYS
IF OD.MFJ(OBSTACLE) GT 0
   SUBTRACT OD.MFJ(OBSTACLE) FROM AVAILABLE.MFJ
ALWAYS
IF OD.M180(OBSTACLE)  GT 0
   SUBTRACT OD.M180(OBSTACLE) FROM AVAILABLE.M180
ALWAYS
RETURN
END
```

```
ROUTINE FIGHT THE.BATTLE GIVEN POINTER, UNIT YIELDING OBSTACLE.USED,
        TEMP.TIME
  DEFINE POINTER, UNIT  AS INTEGER VARIABLES
  DEFINE TEMP.TIME AS A REAL VARIABLE
''*****************************************************************************
''*                                                                           *
''*    This routine conducts a combat simulation between an attacking        *
''*  battalion and the defender CO/TM that is desired to be placed.          *
''*  The routine calls another routine to determine the best feasible        *
''*  obstacle  available for the candidate arc before determining the        *
''*  attacker's deployment rate.  The delay time of the arc, composed of     *
''*  the battle duration time and delay caused by attrition, is passed to    *
''*  the calling routine.                                                     *
''*                                                                           *
''*    NAME: J R MCLAUGHLIN                          DATE: 23 JAN 86          *
''*                                                                           *
''*    CALLED BY MAIN                                                         *
''*                                                                           *
''*****************************************************************************
DEFINE DCEP, DIFCAS, AIFCAS, DROF, AROF, DHCSRT, AHCSRT, DFHEL, ATHEL, DRNALO, ARNALO,
  DHLTM, AHLTM AS REAL VARIABLES
DEFINE VDPDOC, LDOC, WDOC, R, ATRTOT, DEFTOT, CASDLY AS REAL VARIABLES
DEFINE ILOOP, ITMST1, ITMST2, CNT, CNTAA, CNTAD, CNTHA, CNTHD AS INTEGER VARIABLES
DEFINE DEL, ARTAT, DARTAT, AHLCAS, DHLCAS, DTMHA1, DTMHD1, DTMAA1, DTMAD1,
       ADFCAS, DDFCAS, DTM1, DTM2 AS REAL VARIABLES
DEFINE WDAVL, VALFLR, VDEP, TOTFLR, DEFBP, ATRBP, ATRFOR, ATOTAT,
     - DTOTAT, DATR, AATR, ARCDLY, UNIT.TO.PLACE,
       ATATR, ALPHA, BETA, TMSTP1, TMSTP2 AS REAL   -
       VARIABLES
DEFINE DPER, APER, ATDPLYD,
       DEFFOR, DEFATR AS REAL VARIABLES
DEFINE OBSTACLE.USED,
       CNTR, DELTAT, IDFLAG, IAFLAG, J, ARC  AS INTEGER VARIABLES
DEFINE ARC.LABEL AS A TEXT VARIABLE
USE UNIT 16 FOR OUTPUT
LET DCEP = .08                         ''EFFECTS RADIUS OF DEFENDER BATTERY ONE
LET DIFCAS = .20 LET AIFCAS = .005     ''ATTRITION COEFS FOR DEF & ATK ARTY
LET AROF = 3 LET DROF = 3              ''ARTY RATES OF FIRE
LET DHCSRT = 5 LET AHCSRT = 3          ''ATK HELO CAS RATE (SUA/MIN)
LET DFHEL= 16 LET ATHEL = 20           ''NO ATK HELOS AVAILABLE
LET DHLTM = 30  LET AHLTM = 25         ''EFFECTIVE MISSION DURATION OF ATK HELO
LET DRNALO = 30 LET ARNALO = 90        ''ROUND ALLOCATION PER TUBE
LET CNTR = 0
LET TEMP.TIME = 0
LET DPER = .30                         ''DFNDR BREAKPOINT
LET APER = .50                         ''ATTKR BREAKPOINT
LET R= 1500                            ''BN EQUIVALENT TO SUA CONVERSION FCTR
LET ATRTOT = 1500                      ''VALUE OF ATKR BN
LET WDOC = .8                          ''DOCTRINAL DEPLOYMENT WIDTH OF BN
LET LDOC = 1.8                         ''DOCTRINAL DEPLOYMENT LENGTH OF BN
LET VDPDOC = 10                        ''DEPLOYED VELOCITY OF ATKR
LET DELTAT = 2                         ''MINUTES PER BATTLE TIMESTEP
LET WEISS = .7                         ''WEISS PARAMETER FOR HUMBOLDT EQN
LET ALPHA =  1-WEISS                   ''DFNDR WEISS PARAMETER
LET BETA = 1-WEISS                     ''ATKR WEISS PARAMETER
LET DATR = .01                         ''DFNDR ATTRITION COEF
LET AATR = .025                        ''ATTKR ATTRITION COEF
LET CASDLY = 2.5                       ''CASUALTY DELAY PER % ATTKR KILLED (MIN)
   IF RA.NUMBER.LANES.ON.ROUTE(POINTER) EQ 0  ''DETERMINE WDAVL FOR DEPLOYMENT
      IF RA.OFF.ROUTE.CLASS(POINTER) LE 3     ''OF ATTKR ON ARC
```

```
        IF RA.MAIN.ROUTE.CLASS(POINTER) EQ 7
          LET WDAVL = 0
        ELSE
          LET WDAVL = RA.WIDTH.KM(POINTER)
        ALWAYS
    ALWAYS
ALWAYS
IF RA.NUMBER.LANES.ON.ROUTE(POINTER) GT 0
    IF RA.OFF.ROUTE.CLASS(POINTER) LE 3
       LET WDAVL = RA.WIDTH.KM(POINTER) + RA.NUMBER.LANES.ON.ROUTE(POINTER)
                          * .005
    ELSE
         LET WDAVL = RA.NUMBER.LANES.ON.ROUTE(POINTER)* .005
    ALWAYS
ALWAYS

LET ARC.LABEL = TEXT.ARC.TYPES(RA.MAIN.ROUTE.CLASS(POINTER))
LET UNIT.TO.PLACE = FU.SUA(UNIT)
PRINT 5 LINES WITH ARC.LABEL, WDAVL,
      RA.LENGTH.KM(POINTER), POINTER THUS
           TYPE ARC                                               ************
           HAS DEPLOYABLE WIDTH OF                             ****.** KM
           LENGTH OF                                           ****.** KM
           POINTER =                                          ********
IF RA.SPEED.BASIC(POINTER) = 0
   LET RA.SPEED.BASIC(POINTER) = 10
ALWAYS
''       PRINT 1 LINE WITH RA.SPEED.BASIC.(POINTER) THUS
''             ARC SPEED =                                       ***.** KM/HR
IF RA.SPEED.BASIC(POINTER) LT VDPDOC
     LET VDEP = RA.SPEED.BASIC(POINTER)
ELSE
     LET VDEP = VDPDOC
ALWAYS
     PRINT 1 LINE WITH VDEP THUS
           DEPLOYABLE VELOCITY IS                              ****.** KM/HR
LET VALFLR = ((VDEP*WDAVL)/(LDOC*WDOC))*R  ''ARC FLOW RATE W/NO OBSTACLES
IF VALFLR = 0
         LET VALFLR = ATRTOT/2  ''UNIT BRAKES BRUSH AT A HALF MPHR
ALWAYS
PRINT 1 LINE WITH VALFLR/60 THUS
           UNRESTRICTED FLOW RATE FOR THIS ARC IS          ******.** SUA/MIN
                                      ''FIND BEST FEASIBLE OBSTACLE
CALL SELECT.OBSTACLE GIVEN POINTER YIELDING ARCDLY, OBSTACLE.USED
IF OBSTACLE.USED = 0
           PRINT 1 LINE THUS
     XXXXXX ENGINEER TABLE PROVIDES NO OBSTACLE FOR THIS TYPE ARC XXXXX
ELSE
           PRINT 2 LINES WITH ARCDLY, OD.OBSTACLE.NAME(OBSTACLE.USED) THUS
           FOR THIS ARC BEST OBST DELAY IS                      ***.** MINUTES
           WITH OBSTACLE TYPE =                                    **********
ALWAYS
                                      ''FIND ARC FLOW RATE W/BEST POTENTIAL
                                      ''& CURRENT OBSTACLES ON ARC
LET TOTFLR = VALFLR/(60 + ARCDLY + RA.OBSTACLE.DELAY.TIME(POINTER))
PRINT 2 LINES WITH RA.OBSTACLE.DELAY.TIME(POINTER), TOTFLR THUS
           PREVIOUS OBSTACLE DELAY ON ARC =                    ****.** MINUTES
           MAX FLOW RATE WITH PRIOR & 1 NEW OBST IS            ****.** SUA/MIN
LET TMSTP1 = ATRTOT/(TOTFLR*DELTAT)  ''TIMESTEPS TO DEPLOY ATTKR ON ARC
LET ITMST1 = INT.F(TMSTP1) + 1
```

```
PRINT 1 LINE WITH TMSTP1 THUS
         NO. OF STEPS TO DEPLOY ATTACKER ON ARC IS            ***.*

LET TMSTP2 = ((RA.LENGTH.KM(POINTER)*30)/VDEP)/DELTAT ''TIMESTEPS TO
LET ITMST2 = INT.F(TMSTP2) + 1                          ''REACH DFNDR
PRINT 1 LINE WITH TMSTP2 THUS
         NO. OF STEPS TO MOVE FORCE DOWN ARC IS               ****.*
LET DEFTOT = RA.TOTAL.SUA(POINTER)
 PRINT 3 LINES WITH RA.TOTAL.SUA(POINTER),UNIT.TO.PLACE
THUS
         NO. SUA CURRENTLY ON ARC =                           ****.**
         NO. SUA TO PLACE THIS CYCLE =                        ****.**
 IF UNIT GT 0
   ADD  FU.SUA(UNIT) TO DEFTOT  ''VALUE OF DFNDR = CURRENT UNIT &
ALWAYS                          ''OTHERS ALREADY ON ARC
LET DTM1 = TMSTP1 - ITMST1 + 1
LET DTM2 = TMSTP2 - ITMST2 + 1
LET CNTAD = INT.F((DRNALO/DROF)/DELTAT)
LET CNTAA = INT.F((ARNALO/AROF)/DELTAT)
LET CNTHD = INT.F((DHLTM/DELTAT)*3)
LET CNTHA = INT.F(AHLTM/DELTAT)
LET DTMAD1 = (DRNALO/DROF)/DELTAT - CNTAD + 1
LET DTMAA1 = (ARNALO/AROF)/DELTAT - CNTAA + 1
LET DTMHD1 = (DHLTM/DELTAT)*3 - CNTHD + 1
LET DTMHA1 = (AHLTM/DELTAT) - CNTHA + 1
LET ATRFOR=0
LET ATOTAT = 0
LET DTOTAT = 0
LET DEFFOR = DEFTOT
LET ARTAT = 0
LET DARTAT = 0
LET DEFATR = 0
LET ATATR = 0
LET DEFBP = DPER*DEFTOT                       ''DFNDR BREAKPOINT SUA
LET ATRBP = APER*ATRTOT                       ''ATTKR BREAKPOINT SUA
''  PRINT 4 LINES WITH DTM1,DTM2,CNTAA,CNTAD,CNTHA,CNTHD,DTMAA1,DTMAD1,
''           DTMHA1,DTMHD1 THUS
''
''  DTM1  DTM2  CNTAA CNTAD  CNTHA  CNTHD DTMAA1 DTMAD1 DTMHA1 DTMHD1
''  *.**  *.**   ***   ****   ***    ***  **.**  **.**  **.**  **.**
''
    PRINT 6 LINES WITH DATR,AATR,(1-ALPHA),(1-BETA),
           AIFCAS,DIFCAS,AHCSRT,DHCSRT THUS

         DFNDR     ATKR    DEF     ATKR      AIFCAS   DIFCAS  AHL    DH
         ATRT      ATRT    WEISS   WEISS                      CSRT   CSRT
         COEF      COEF
         *.***     *.***   *.***   *.***     **.**    **.**   **.**  **.**

    PRINT 5 LINE WITH DEFBP,ATRBP THUS
         DEF BREAKPOINT IS ****.*               ATKR BREAKPOINT IS  ****.*

  STP  ATKR  ATTKR  DFNDR  ATKR  DFDR  ATTKR  DFDR ATTKR DFDR  ATKTO  DFTOT
       DPLYD ONARC  ONARC  DFCS  DFCS  HLCAS  HLCS ARTCS ARTC  CAS    CAS
 LET IDFLAG = 0                        ''FLAG INDICATING DFNDR BROKE
 LET IAFLAG = 0                       ''FLAG INDICATING ATTKR BROKE
''FIGHT THE BATTLE ON CURRENT ARC WHILE ATKR DEPLOYS
 LET ILOOP = ITMST1 + ITMST2          ''BATTLE MAX DURATION
 FOR J = 1 TO ILOOP
   DO
```

```
      LET CNT = J
      LET DEL = DELTAT
      IF CNT = ILOOP
         LET DEL = DTM2
      ALWAYS
      IF CNT LE ITMST1                              ''DEPLOY MORE ATTKRS
         IF CNT = ITMST1                            ''IF NOT ALREADY DEPLOYED
            LET DEL = DTM1
         ALWAYS
         LET ATRFOR = ATRFOR + (TOTFLR*DEL)
         ADD TOTFLR*DEL TO ATDPLYD
      ALWAYS
                                                    ''ATTACKER ATTRITION FROM DEF ARTY
      LET ARTAT = 0
''       LET DEL = DELTAT
''       IF CNT LE CNTAD + 1
''          IF CNT = CNTAD + 1
''             LET DEL = DTMAD1
''          ALWAYS
''          LET ARTAT = (ATRFOR/(LDOC*WDAVL))*DCEP*DIFCAS*DROF*DEL
''       ALWAYS
                                                    ''DEF ENDER ATTRITION FROM ATTKR ARTY
      LET DARTAT = 0
''          IF CNT LE CNTAA + 1
''            _IF CNT EQ CNTAD + 1
''                LET DEL = DTMAA1
''             ALWAYS
''             LET DARTAT = DEFFOR*AIFCAS*AROF*DEL
''          ALWAYS
                                                 ''ATTACKER ATTRITION FROM DFNDR HELOS
       LET AHLCAS = 0
    ''    LET DEL = DELTAT
    ''    IF CNT LE CNTHD GO TO ATKRHELOCAS ELSE
    ''    IF CNT = CNTHD + 1 LET DEL = DTMHD1 GO TO ATKRHELOCAS ELSE
    ''      GO TO NODEFHELOCAS
    ''    'ATKRHELOCAS'
    ''    LET AHLCAS = (DFHEL/3)*DHCSRT*DEL
                                                    ''DEFENDER ATTRITION FROM ATTKR HELOS
     'NODEFHELOCAS'
         LET DHLCAS = 0
    ''   IF CNT LE CNTHA GO TO DEFHELOCAS ELSE
    ''    IF CNT = CNTR + 1 LET DEL = DTMHA1 GO TO DEFHELOCAS ELSE
    ''      GO TO NOATKHELOS
    ''     'DEFHELOCAS'
    ''        LET DHLCAS = ATHEL*AHCSRT*DEL
    ''     'NOATKHELOS'
                                                    ''DIRECT FIRE ATTRITION
       LET DDFCAS = DATR*ATRFOR*((DEFFOR/ATRFOR)**ALPHA)*DELTAT
       IF DEFFOR = 0
               LET ADFCAS = 0
       ELSE
               LET ADFCAS = AATR*DEFFOR*((ATRFOR/DEFFOR)**BETA)*DELTAT
       ALWAYS
                                                    ''TOTAL ATTRITION EQNS
       LET DEFATR = DDFCAS + DHLCAS + DARTAT
       LET ATATR = ADFCAS + AHLCAS + ARTAT
       IF DEFATR GT DEFFOR
                LET DEFATR = DEFFOR .
       ALWAYS
       IF ATATR GT ATRFOR
```

```
            LET ATATR = ATRFOR
    ALWAYS
                                                ''SUBTRACT ATTRITION FROM
    LET ATOTAT = ATOTAT + ATATR                 ''FORCES ON ARC
    LET DTOTAT = DTOTAT + DEFATR
     PRINT 1 LINE WITH J, ATDPLYD, ATRFOR, DEFFOR, ADFCAS, DDFCAS, AHLCAS, DHLCAS,
           ARTAT, DARTAT, ATOTAT, DTOTAT THUS
  ***   ****.* ****.*   ***.*   ***.* **.*   ***.*   **.* ***.* **.*   ****.* ***.*
     LET DEFFOR = DEFFOR-DEFATR
    LET ATRFOR = ATRFOR-ATATR
                                               ''CHECK FOR BREAKPOINT
                                               ''DURING THE PREVIOUS TIMESTEP
      IF DEFFOR LE DEFBP
         LET IDFLAG = 1
      ALWAYS
      IF (ATRTOT - ATOTAT) LE ATRBP
         LET IAFLAG = 1
      ALWAYS
      IF IDFLAG = 0 AND IAFLAG = 0
         CYCLE
      OTHERWISE
      IF IDFLAG = 1
         LET TEMP.TIME = J*DELTAT +
                                CASDLY*((100*ATOTAT)/ATRTOT)
                PRINT 3 LINES WITH (J), TEMP.TIME THUS

    XXXX   DEFENSE BROKE AT TIMESTEP ******** NEW ARC TIME IS *****.** MINUTES

       ALWAYS
      IF IAFLAG = 1
         LET TEMP.TIME = 999

              PRINT 3 LINES WITH (J), TEMP.TIME THUS

XXXX   ATTACKER BROKE AT TIMESTEP ******  NEW ARC TIME IS *****.** MIN

       ALWAYS
      IF IAFLAG = 1 OR IDFLAG = 1
         EXIT
      OTHERWISE
   LOOP
   LET TEMP.TIME = J*DELTAT +
                         CASDLY*((100*ATOTAT)/ATRTOT)
    PRINT 3 LINES WITH TEMP.TIME THUS

 XXXX NEITHER SIDE BREAKS   NEW ARC TIME IS *******.**   MIN XXXX

   'OUT'
  RETURN
  END     ''OF ROUTINE DEPLOY.ASSETS.IN.SECTOR
```

```
ROUTINE SELECT.OBSTACLE
GIVEN POINTER
YIELDING ARCDLY,OBSTACLE.USED
''******************************************************************
''*                                                                *
''*  THIS ROUTINE FINDS THE MOST DELAY-PRODUCING, FEASIBLE         *
''*  OBSTACLE THAT IS AVAILABLE FOR AN ARC.  THE DELAY TIME        *
''*  AND THE OBSTACLE TYPE ARE PASSED BACK TO THE CALLING          *
''*  ROUTINE.  THE OBSTACLE BANK ACCOUNT IS NOT ALTERED.           *
''*                                                                *
''*  NAME: J MCLAUGHLIN              DATE: 15 JAN 86               *
''*                                                                *
''*                                                                *
''******************************************************************
''DEFINE LOCAL VARIABLES
  DEFINE POINTER,TYPE.ARC,POSSIBLE.TYPE,TYPE.OBSTACLE,OBSTACLE.USED
   AS INTEGER VARIABLES
  DEFINE ARCDLY AS A REAL VARIABLE
LET TYPE.ARC =  RA.MAIN.ROUTE.CLASS(POINTER)
FOR EACH POSSIBLE.TYPE IN RC.APPLICABLE.OBSTACLE.SET(TYPE.ARC)
    DO
     IF POSSIBLE.TYPE = 0
       GO TO NOBST
     ALWAYS
       ''CHECK TO SEE IF ASSETS FOR OBSTACLE ARE AVAILABLE
       LET TYPE.OBSTACLE = AO.OBSTACLE.CLASS(POSSIBLE.TYPE)
       IF OD.SQUAD.HOURS(TYPE.OBSTACLE) GT 0
          IF AVAILABLE.SQUAD.HOURS LT OD.SQUAD.HOURS(TYPE.OBSTACLE)
          ''NOT FEASIBLE
            GO TO CHECK.NEXT.OBSTACLE
          ALWAYS
       ALWAYS
       IF OD.DOZER.HOURS(TYPE.OBSTACLE) GT 0
          IF AVAILABLE.DOZER.HOURS LT OD.DOZER.HOURS(TYPE.OBSTACLE)
              ''NOT FEASIBLE
             GO TO CHECK.NEXT.OBSTACLE
          ALWAYS
       ALWAYS
       IF OD.BRIDGE.DEMO.KITS(TYPE.OBSTACLE) GT 0
          IF AVAILABLE.BRIDGE.DEMO.KITS LT OD.BRIDGE.DEMO.KITS(TYPE.OBSTACLE)
             ''NOT FEASIBLE
             GO TO CHECK.NEXT.OBSTACLE
          ALWAYS
       ALWAYS
       IF OD.M180(TYPE.OBSTACLE) GT 0
          IF AVAILABLE.M180 LT OD.M180(TYPE.OBSTACLE)
             ''NOT FEASIBLE
             GO TO CHECK.NEXT.OBSTACLE
          ALWAYS
       ALWAYS
       IF OD.MFJ(TYPE.OBSTACLE) GT 0
          IF AVAILABLE.MFJ LT OD.MFJ(TYPE.OBSTACLE)
             ''NOT FEASIBLE
             GO TO CHECK.NEXT.OBSTACLE
          ALWAYS
       ALWAYS
       IF OD.FUEL.GALLONS(TYPE.OBSTACLE) GT 0
          IF AVAILABLE.FUEL.GALLONS LT OD.FUEL.GALLONS(TYPE.OBSTACLE)
             ''NOT FEASIBLE
             GO TO CHECK.NEXT.OBSTACLE
```

```
            ALWAYS
         ALWAYS
         IF OD.MOPHS(TYPE.OBSTACLE) GT 0
            IF AVAILABLE.MOPHS LT OD.MOPHS(TYPE.OBSTACLE)
               ''NOT FEASIBLE
               GO TO CHECK.NEXT.OBSTACLE
            ALWAYS
         ALWAYS
         LET ARCDLY = AO.DELAY(POSSIBLE.TYPE)
         LET OBSTACLE.USED  = AO.OBSTACLE.CLASS(POSSIBLE.TYPE)
        RETURN
'NOBST' LET ARCDLY = 0
          LET OBSTACLE.USED = 0
          RETURN
'CHECK.NEXT.OBSTACLE'
      LOOP
     END
```

```
ROUTINE DEPLOY ASSETS. IN. SECTOR
   GIVEN SECTOR, AVE. OF. APPR
   DEFINE SECTOR, AVE. OF. APPR AS INTEGER VARIABLES
''*****************************************************************************
''*                                                                           *
''*    This routine calls the routines to find the most dangerous avenue of   *
''*  approach, pick an arc on the avenue to block, deploy the force or        *
''*  obstacles  or both, and continue until there are no more forces or       *
''*  obstacles left to deploy.                                                *
''*                                                                           *
''*    NAME: J R MCLAUGHLIN                         DATE: 15 JAN 86           *
''*                                                                           *
''*    CALLED BY MAIN                                                         *
''*                                                                           *
''*****************************************************************************
''FIRST DEFINE LOCAL VARIABLES
DEFINE WIDTH, FLOWRT, X1, X, Y, X2, Y1, Y2 AS REAL VARIABLES
DEFINE COUNT, BEST. TIME. POINTER, AS. COUNTER AS INTEGER VARIABLES
DEFINE ARCDLY, TEMP. TIME AS REAL VARIABLES
DEFINE BEST. TIME, BEST. ARC. OBSTACLE. DELAY. TIME AS REAL VARIABLES
DEFINE BEST. ARC. OBSTACLE. TYPE AS INTEGER VARIABLES
DEFINE UNIT, NR. IN. LIST AS A INTEGER VARIABLE
DEFINE OBSTACLE, OBSTACLE. USED, ARC. SRG, I, POINTER AS INTEGER VARIABLES
''   VARIABLES FOR. SHIFTING FROM ARC TO AROUND A NODE
DEFINE TEST. ARC, RECEIVING. ARC, BEST. ARC, END. 1, END. 2, THE. NODE, THE. FORCE
       AS INTEGER VARIABLES
DEFINE THE. SUA, MOVED. SUA
       AS REAL VARIABLES
''   VARIABLES FOR DETERMINING WEIGHTING FUNCTION
DEFINE FLOT. SLOPE, FLOT. MIDPT. X, FLOT. MIDPT. Y, REAR. MIDPT. X, REAR. MIDPT. Y,
       LAST. Y. INTERCEPT, LAST. MIDPT. X, LAST. MIDPT. Y,
       FIRST. MIDPT. X, FIRST. MIDPT. Y,
       SLOPE, INTERCEPT, MIDLINE. SLOPE, FIRST. Y. INTERCEPT,
       DISTANCE, WEIGHT. DIVISOR, WEIGHT. NUMERATOR,
       WT. FNC, ARC. MIDPT. Y, ARC. MIDPT. X, THE. X, THE. Y, BISECTOR. INTERCEPT
       AS REAL VARIABLES
DEFINE THIS. SEGMENT, THAT. SEGMENT, CURRENT. ARC,
       ANSWER, REAR. BASELINE, BISECTOR AS INTEGER VARIABLES
  USE UNIT 15 FOR OUTPUT
''NEXT IF ANY FORCES AVAILABLE, GRAB ONE
  FOR EACH UNIT IN US. SUBORDINATE. FORCE. LIST(SECTOR)
     WITH UNIT NOT IN SOME RA. FORCE. UNIT. SET
     AND UNIT NOT IN SOME RN. FORCE. UNIT. SET
        FIND THE FIRST CASE
LET COUNT = 0
PRINT 5 LINES THUS

ARC      ARC          NO.  OFF-RT  MN-RT  OFF-RT LENGTH  COST    FLOW RT    POINTER
NO.      TYPE         LNS  WD (KM) CLASS  CLASS   KM)             (SUA/MIN)


''COMPLETE TABLE LISTED ABOVE FOR EACH ARC IN THE PATH
  FOR EACH ARC. SRG IN ADA. ROUTE(AVE. OF. APPR)
  DO
     IF RA. DUMMY(AS. ARC. POINTER(ARC. SRG)) IS. EQUAL YES
        CYCLE
     OTHERWISE
     LET POINTER= AS. ARC. POINTER(ARC. SRG)
     ADD 1 TO COUNT
''DETERMINE WIDTH AVAILABLE FOR ATTACKER DEPLOYMENT
```

```
      IF RA.NUMBER LANES.ON.ROUTE(POINTER) EQ 0
         IF RA.OFF ROUTE.CLASS(POINTER) LE 3
            IF RA.MAIN.ROUTE.CLASS(POINTER) EQ 7
               LET WIDTH = 0
            ELSE
               LET WIDTH = RA.WIDTH.KM(POINTER)
            ALWAYS
         ALWAYS
      ALWAYS
      IF RA.NUMBER.LANES.ON.ROUTE(POINTER) GT 0
         IF RA.OFF.ROUTE.CLASS(POINTER) LE 3
            LET WIDTH = RA.NUMBER.LANES.ON.ROUTE(POINTER) * .005
               + RA.WIDTH.KM(POINTER)
         ELSE
            LET WIDTH = RA.NUMBER.LANES.ON.ROUTE(POINTER) * .005
         ALWAYS
      ALWAYS
      LET FLOWRT = (((10 * WIDTH)/(.8 * 1.8)*1500)/60)
      PRINT 1 LINE WITH COUNT,TEXT.ARC.TYPES(RA.MAIN.ROUTE.CLASS(POINTER)),
            RA.NUMBER.LANES.ON.ROUTE(POINTER),RA.WIDTH.KM(POINTER),
            RA.MAIN.ROUTE.CLASS(POINTER),
            RA.OFF.ROUTE.CLASS(POINTER),RA.LENGTH.KM(POINTER),RA.COST(POINTER),
            FLOWRT, POINTER THUS
**  ************    **   **.**    **     **    **.**   ***.**  ***.**   *******
   LOOP
''DEVELOP WEIGHTING SCHEME FOR ARC PROXIMITY TO THE FLOT
   LET FLOT.SLOPE = LS.SLOPE(US.FLOT.LS(SECTOR))
   LET FLOT.MIDPT.X = BP.X(US.FLOT.POINT.1(SECTOR)) +
       (BP.X(US.FLOT.POINT.2(SECTOR)) - BP.X(US.FLOT.POINT.1(SECTOR)))*.5
   LET FLOT.MIDPT.Y = BP.Y(US.FLOT.POINT.1(SECTOR)) +
       (BP.Y(US.FLOT.POINT.2(SECTOR)) - BP.Y(US.FLOT.POINT.1(SECTOR)))*.5
   LET REAR.MIDPT.X = BP.X(US.REAR.POINT.1(SECTOR)) +
       (BP.X(US.REAR.POINT.2(SECTOR)) - BP.X(US.REAR.POINT.1(SECTOR)))*.5
   LET REAR.MIDPT.Y = BP.Y(US.REAR.POINT.1(SECTOR)) +
       (BP.Y(US.REAR.POINT.2(SECTOR)) - BP.Y(US.REAR.POINT.1(SECTOR)))*.5
   LET X1 = FLOT.MIDPT.X
   LET X2 = REAR.MIDPT.X
   LET Y1 = FLOT.MIDPT.Y
   LET Y2 = REAR.MIDPT.Y
   CALL STANDARD.EQUATION GIVEN X1,Y1,X2,Y2 YIELDING SLOPE,INTERCEPT'
   USE THE.TERMINAL FOR OUTPUT
   LET MIDLINE.SLOPE = SLOPE
PRINT 7 LINES WITH FLOT.SLOPE,FLOT.MIDPT.X,FLOT.MIDPT.Y,REAR.MIDPT.X,
      REAR.MIDPT.Y, MIDLINE.SLOPE  THUS


FLOT       FLOT       FLOT       REAR       REAR       REAR MIDLINE
SLOPE      MIDPT X    MIDPT Y    MIDPT X    MIDPT Y    SLOPE
**.**      ***.*      ***.*      ***.*      ***.*      **.**

   LET ARC.SRG =  P.ADA.ROUTE(L.ADA.ROUTE(AVE.OF.APPR))
   LET POINTER = AS.ARC.POINTER(ARC.SRG)
   LET X1 = RN.EASTING(RA.END.1.NODE(POINTER))
   LET Y1 = RN.NORTHING(RA.END.1.NODE(POINTER))
   LET X2 = RN.EASTING(RA.END.2.NODE(POINTER))
   LET Y2 = RN.NORTHING(RA.END.2.NODE(POINTER))
   LET LAST.MIDPT.X = X1 + (X2 - X1)*.5
   LET LAST.MIDPT.Y = Y1 + (Y2 - Y1)*.5
   LET X = LAST.MIDPT.X
```

```
  LET Y = LAST.MIDPT.Y
  LET SLOPE = FLOT.SLOPE
  CALL SLOPE.POINT.INTERCEPT GIVEN SLOPE,X,Y YIELDING INTERCEPT
  LET LAST.Y.INTERCEPT = INTERCEPT
  USE THE.TERMINAL FOR OUTPUT
  PRINT 1 LINE WITH LAST.MIDPT.X,LAST.MIDPT.Y,LAST.Y.INTERCEPT THUS
 LAST.MIDPT.X = ****.* LAST.MIDPT.Y = ****.* LAST.Y.INTERCEPT = ****.*
  CREATE A LINE.SEGMENT CALLED REAR.BASELINE
     LET LS.X1(REAR.BASELINE) = RINF.C
     LET LS.Y1(REAR.BASELINE) = RINF.C
     LET LS.X2(REAR.BASELINE) = -RINF.C
     LET LS.Y2(REAR.BASELINE) = -RINF.C
     LET LS.SLOPE(REAR.BASELINE) = FLOT.SLOPE
     LET LS.INTERCEPT(REAR.BASELINE) = LAST.Y.INTERCEPT
  FOR EACH ARC.SRG IN AOA.ROUTE(AVE.OF.APPR) '' FIND FIRST ARC MIDPOINT
  DO
     IF AS.FEBA.OR.REAR(ARC.SRG) IS.NOT.EQUAL FEBA
        CYCLE
     OTHERWISE
        LET POINTER= AS.ARC.POINTER(ARC.SRG)
        LET X1 = RN.EASTING(RA.END.1.NODE(POINTER))
        LET Y1 = RN.NORTHING(RA.END.1.NODE(POINTER))
        LET X2 = RN.EASTING(RA.END.2.NODE(POINTER))
        LET Y2 = RN.NORTHING(RA.END.2.NODE(POINTER))
        LET FIRST.MIDPT.X = X1 + (X2 - X1)*.5
        LET FIRST.MIDPT.Y = Y1 + (Y2 - Y1)*.5
        LET X = FIRST.MIDPT.X
        LET Y = FIRST.MIDPT.Y
        LET SLOPE = MIDLINE.SLOPE
        CALL SLOPE.POINT.INTERCEPT GIVEN SLOPE,X,Y YIELDING INTERCEPT
        USE THE.TERMINAL FOR OUTPUT
        PRINT 1 LINE WITH INTERCEPT,SLOPE,X,Y THUS
BISECTOR Y AXIS INTERCEPT = ****.* INPT SLOPE = ****.* X= ***.* Y = ***.*
        LET BISECTOR.INTERCEPT = INTERCEPT
        CREATE A LINE.SEGMENT CALLED BISECTOR
        LET LS.X1(BISECTOR) = RINF.C
        LET LS.Y1(BISECTOR) = RINF.C
        LET LS.X2(BISECTOR) = -RINF.C
        LET LS.Y2(BISECTOR) = -RINF.C
        LET LS.SLOPE(BISECTOR) = MIDLINE.SLOPE
        LET LS.INTERCEPT(BISECTOR) = BISECTOR.INTERCEPT
        LET THIS.SEGMENT = BISECTOR
        LET THAT.SEGMENT = REAR.BASELINE
        CALL DO.TWO.SEGMENTS.INTERSECT GIVEN THIS.SEGMENT,THAT.SEGMENT
             YIELDING ANSWER,THE.X,THE.Y
        LET X1 = THE.X
        LET Y1 = THE.Y
        LET X2 = FIRST.MIDPT.X
        LET Y2 = FIRST.MIDPT.Y
        CALL FIND.SEGMENT.LENGTH GIVEN X1,Y1,X2,Y2 YIELDING DISTANCE
        LET WEIGHT.DIVISOR = DISTANCE
  LOOP
USE THE.TERMINAL FOR OUTPUT
PRINT 5 LINES WITH THE.X,THE.Y,WEIGHT.DIVISOR THUS

    FIRST ARC          FIRST ARC         DISTANCE
    REAR X INT         REAR Y INT        TO REAR
      ****.*             ****.*           ***.*

''CALCULATE WEIGHT FNC FOR EACH ARC ON PATH
```

```
USE UNIT 15 FOR OUTPUT
PRINT 5 LINES THUS


ARC      ARC      ARC      ARC      ARC      WEIGHT    DELAY    OBJ       POINTER
REARX    REARY    MIDX     MIDY     LENGTH   FACTOR    TIME     TIME

  FOR EACH ARC.SRG IN AOA.ROUTE(AVE.OF.APPR)
  DO
     IF AS.FEBA.OR.REAR(ARC.SRG) IS.NOT.EQUAL FEBA AND
        AS.FEBA.OR.REAR(ARC.SRG) IS.NOT.EQUAL REAR AND
        AS.ARC.CLASS(ARC.SRG) IS.NOT.EQUAL BORDER.CROSSER AND
        AS.ARC.CLASS(ARC.SRG) IS.NOT.EQUAL COMPLETELY.IN
        CYCLE
     OTHERWISE
     LET POINTER= AS.ARC.POINTER(ARC.SRG)
        LET X1 = RN.EASTING(RA.END.1.NODE(POINTER))
        LET Y1 = RN.NORTHING(RA.END.1.NODE(POINTER))
        LET X2 = RN.EASTING(RA.END.2.NODE(POINTER))
        LET Y2 = RN.NORTHING(RA.END.2.NODE(POINTER))
        LET ARC.MIDPT.X = X1 + (X2 - X1)*.5
        LET ARC.MIDPT.Y = Y1 + (Y2 - Y1)*.5
        LET X = ARC.MIDPT.X
        LET Y = ARC.MIDPT.Y
        LET SLOPE = MIDLINE.SLOPE
        CALL SLOPE.POINT.INTERCEPT GIVEN SLOPE,X,Y YIELDING INTERCEPT
        CREATE A LINE.SEGMENT CALLED CURRENT.ARC
        LET LS.X1(CURRENT.ARC) = RINF.C
        LET LS.Y1(CURRENT.ARC) = RINF.C
        LET LS.X2(CURRENT.ARC) = -RINF.C
        LET LS.Y2(CURRENT.ARC) = -RINF.C
        LET LS.SLOPE(CURRENT.ARC) = MIDLINE.SLOPE
        LET LS.INTERCEPT(CURRENT.ARC) = INTERCEPT
        LET THIS.SEGMENT = CURRENT.ARC
        LET THAT.SEGMENT = REAR.BASELINE
        CALL DO.TWO.SEGMENTS.INTERSECT GIVEN THIS.SEGMENT,THAT.SEGMENT
           YIELDING ANSWER,THE.X,THE.Y
        LET X1 = THE.X
        LET Y1 = THE.Y
        LET X2 = ARC.MIDPT.X
        LET Y2 = ARC.MIDPT.Y
        CALL FIND.SEGMENT.LENGTH GIVEN X1,Y1,X2,Y2 YIELDING DISTANCE
        LET WEIGHT.NUMERATOR = DISTANCE
 ''DETERMINE THE BATTLE TIME OF THE CURRENT ARC
        CALL FIGHT.THE.BATTLE GIVEN POINTER, UNIT YIELDING
             OBSTACLE.USED,TEMP.TIME
        ''USE THE.TERMINAL FOR OUTPUT
        USE UNIT 15 FOR OUTPUT
        LET WT.FNC = (WEIGHT.NUMERATOR/WEIGHT.DIVISOR) + 1
        PRINT 1 LINE WITH X1,Y1,X2,Y2,WEIGHT.NUMERATOR,
              WT.FNC,TEMP.TIME,TEMP.TIME*WT.FNC,POINTER THUS
***.*    ***.*    ***.*    ***.*    ***.*      *.**    ****.**   ****.**   *******
     LET AS.TIME(ARC.SRG) = TEMP.TIME*WT.FNC
     LET AS.OBSTACLE.TYPE(ARC.SRG) = OBSTACLE.USED
     FILE ARC.SRG IN BEST.TIME.LIST
  LOOP
  LET BEST.TIME.POINTER = AS.ARC.POINTER(F.BEST.TIME.LIST)
  LET BEST.TIME = AS.TIME(F.BEST.TIME.LIST)
  LET BEST.ARC.OBSTACLE.TYPE = AS.OBSTACLE.TYPE(F.BEST.TIME.LIST)
  IF BEST.ARC.OBSTACLE.TYPE GT  0
```

```
   LET BEST ARC.OBSTACLE.DELAY.TIME = OD.DELAY.CAUSED(BEST.ARC.OBSTACLE.TYPE)
ELSE
   LET BEST.ARC.OBSTACLE.DELAY.TIME = 0
ALWAYS
CREATE AN ENGINEER.OBSTACLE CALLED OBSTACLE
LET EO.OBSTACLE.CLASS(OBSTACLE) = BEST.ARC.OBSTACLE.TYPE
LET EO.RA.POINTER.(OBSTACLE) = BEST.TIME.POINTER
FILE OBSTACLE IN US.OBSTACLE.LIST(SECTOR)
FILE OBSTACLE IN RA.OBSTACLE.SET(BEST.TIME.POINTER)
IF N.BEST.TIME.LIST GT 1
   LET NR.IN.LIST = 2
ELSE
   LET NR.IN.LIST = N.BEST.TIME.LIST
ALWAYS
FOR I = 1 TO NR.IN.LIST
   DO
   LET RA.COLOR.ME(AS.ARC.POINTER(F.BEST.TIME.LIST)) = COLOR.ARRAY(I)
   REMOVE F.BEST.TIME.LIST FROM BEST.TIME.LIST
LOOP
PRINT 3 LINES WITH BEST.TIME,BEST.TIME.POINTER THUS

BEST ARC DELAY = *****.**  AT ARC *********

''THE FOLLOWING LINES DETERMINE IF A PREVIOUSLY PLACED UNIT
''SHOULD BE MOVED FROM AN ARC TO A NODE
''TEST FOR ADJACENCY OF SELECTED ARC TO PREVIOUSLY SELECTED ARC
 LET_END.1 = RA.END.1.NODE(BEST.TIME.POINTER)
 LET BEST.ARC = BEST.TIME.POINTER
 LET END.2 = RA.END.2.NODE(BEST.TIME.POINTER)
''CASE WHERE NEITHER NODE HAS ADJACENT MANNED ARC(S)
 IF RN.FORCE.ADJACENT.TO.NODE(END.1) LT 1
    AND
    RN.FORCE.ADJACENT.TO.NODE(END.2) LT 1
    PRINT 1 LINE THUS
    XXXXXX NO UNITS ON ADJACENT ARCS, GO TO PLACE.UNIT XXXX
    GO TO PLACE.UNIT
 ALWAYS
''CASE WHERE BOTH NODES HAVE ADJACENT MANNED ARCS
 IF RN.FORCE.ADJACENT.TO.NODE(END.1) GT 0
    AND
    RN.FORCE.ADJACENT.TO.NODE(END.2) GT 0
    IF N.RA.FORCE.UNIT.SET(BEST.ARC) GT 0
       PRINT 1 LINE THUS
       XXXXXX  UNIT ALREADY ON ARC, ADD ANOTHER XXXXXX
       GO TO PLACE.UNIT
    OTHERWISE
       PRINT 1 LINE THUS
   XXX UNITS ON ADJACENT ARC(S) OF BOTH END NODES, FIND BEST UNIT TO MOVE XXX
    FOR EACH TEST.ARC IN RN.END.1.NODE.SET(END.1)
        DO
        IF TEST.ARC = BEST.ARC
           CYCLE
        OTHERWISE
        IF RA.FORCE.UNIT.SET(TEST.ARC) IS EMPTY
           CYCLE
        OTHERWISE
        IF FU.SUA(F.RA.FORCE.UNIT.SET(TEST.ARC)) GT THE.SUA
           LET THE.FORCE = F.RA.FORCE.UNIT.SET(TEST.ARC)
           LET THE.SUA = FU.SUA(THE.FORCE)
           LET THE.NODE = END.1
```

```
        LET BEST.ARC = TEST ARC
      ALWAYS
   LOOP
   FOR EACH TEST.ARC IN RN.END.2.NODE.SET(END.1)
      DO
      IF TEST.ARC = BEST.ARC
         CYCLE
      OTHERWISE
      IF RA.FORCE.UNIT.SET(TEST.ARC) IS EMPTY
         CYCLE
      OTHERWISE
      IF FU.SUA(F.RA.FORCE.UNIT.SET(TEST.ARC)) GT THE.SUA
         LET THE.FORCE = F.RA.FORCE.UNIT.SET(TEST.ARC)
         LET THE.SUA = FU.SUA(THE.FORCE)
         LET THE.NODE = END.1
         LET BEST.ARC = TEST.ARC
      ALWAYS
    LOOP
    FOR EACH TEST.ARC IN RN.END.1.NODE.SET(END.2)
       DO
       IF TEST.ARC = BEST.ARC
          CYCLE
       OTHERWISE
       IF RA.FORCE.UNIT.SET(TEST.ARC) IS EMPTY
          CYCLE
       OTHERWISE
       IF FU.SUA(F.RA.FORCE.UNIT.SET(TEST.ARC)) GT THE.SUA
          LET THE.FORCE = F.RA.FORCE.UNIT.SET(TEST.ARC)
          LET THE.SUA = FU.SUA(THE.FORCE)
          LET THE.NODE = END.2
          LET BEST.ARC = TEST.ARC
       ALWAYS
     LOOP
     FOR EACH TEST.ARC IN RN.END.2.NODE.SET(END.2)
        DO
        IF TEST.ARC = BEST.ARC
           CYCLE
        OTHERWISE
        IF RA.FORCE.UNIT.SET(TEST.ARC) IS EMPTY
           CYCLE
        OTHERWISE
        IF FU.SUA(F.RA.FORCE.UNIT.SET(TEST.ARC)) GT THE.SUA
           LET THE.FORCE = F.RA.FORCE.UNIT.SET(TEST.ARC)
           LET THE.SUA = FU.SUA(THE.FORCE)
           LET THE.NODE = END.2
           LET BEST.ARC = TEST.ARC
        ALWAYS
      LOOP
      IF THE.FORCE = 0
         PRINT 1 LINE THUS
         XXXXX NO FORCE FOUND ON ADJACENT ARCS - ERROR- XXXXX
         GO TO PLACE.UNIT
      OTHERWISE
      REMOVE THE.FORCE FROM RA.FORCE.UNIT.SET(BEST.ARC)
      PRINT 2 LINES WITH THE.FORCE,BEST.ARC THUS
      REMOVE ****** SUA FROM ARC **********
      ADD HALF OF THIS TO EACH ADJACENT ARC
      LET RN.FORCE.ADJACENT.TO.NODE(RA.END.1.NODE(BEST.ARC)) = 0
      LET RN.FORCE.ADJACENT.TO.NODE(RA.END.2.NODE(BEST.ARC)) = 0
      FILE THE.FORCE IN RN.FORCE.UNIT.SET(THE.NODE)
```

```
      LET FU.EASTING(THE.FORCE) = RN.EASTING(THE.NODE)
      LET FU.NORTHING(THE.FORCE) = RN.NORTHING(THE.NODE)
      SUBTRACT THE.SUA FROM RA.TOTAL.SUA(BEST.ARC)
      LET MOVED.SUA = .5*THE.SUA
      FOR EACH TEST.ARC IN RN.END.1.NODE.SET(THE.NODE)
         DO
         ADD MOVED.SUA TO RA.TOTAL.SUA(TEST.ARC)
         LET POINTER = TEST.ARC
         CALL SELECT.OBSTACLE GIVEN POINTER YIELDING ARCDLY,OBSTACLE.USED
         IF OBSTACLE.USED GT 0
            ADD ARCDLY TO RA.OBSTACLE.DELAY.TIME(POINTER)
            CREATE AN ENGINEER.OBSTACLE CALLED OBSTACLE
            LET EO.OBSTACLE.CLASS(OBSTACLE) = OBSTACLE.USED
            LET EO.RA.POINTER.(OBSTACLE) = POINTER
            FILE OBSTACLE IN US.OBSTACLE.LIST(SECTOR)
            FILE OBSTACLE IN RA.OBSTACLE.SET(POINTER)
            LET BEST.ARC.OBSTACLE.TYPE = OBSTACLE.USED
            PERFORM ENGINEER.ASSET.UPDATE GIVEN BEST.ARC.OBSTACLE.TYPE
            PRINT 1 LINE WITH OD.OBSTACLE.NAME(OBSTACLE.USED) THUS
            AN *********** OBSTACLE HAS BEEN PLACED ON AN ARC INTO THE NODE
            LET X1 = RN.EASTING(RA.END.1.NODE(POINTER))
            LET Y1 = RN.NORTHING(RA.END.1.NODE(POINTER))
            LET X2 = RN.EASTING(RA.END.2.NODE(POINTER))
            LET Y2 = RN.NORTHING(RA.END.2.NODE(POINTER))
            LET EO.EASTING(OBSTACLE) = X1 + (X2 - X1)*.5
            LET EO.NORTHING(OBSTACLE) = Y1 + (Y2 - Y1)*.5
            ALWAYS
      LOOP
      FOR EACH TEST.ARC IN RN.END.2.NODE.SET(THE.NODE)
         DO
         ADD MOVED.SUA TO RA.TOTAL.SUA(TEST.ARC)
         LET POINTER = TEST.ARC
         CALL SELECT.OBSTACLE GIVEN POINTER YIELDING ARCDLY,OBSTACLE.USED
         IF OBSTACLE.USED GT 0
            ADD ARCDLY TO RA.OBSTACLE.DELAY.TIME(POINTER)
            CREATE AN ENGINEER OBSTACLE CALLED OBSTACLE
            LET EO.OBSTACLE.CLASS(OBSTACLE) = OBSTACLE.USED
            LET EO.RA.POINTER.(OBSTACLE) = POINTER
            FILE OBSTACLE IN US.OBSTACLE.LIST(SECTOR)
            FILE OBSTACLE IN RA.OBSTACLE.SET(POINTER)
            LET BEST.ARC.OBSTACLE.TYPE = OBSTACLE.USED
            PERFORM ENGINEER.ASSET.UPDATE GIVEN BEST.ARC.OBSTACLE.TYPE
            PRINT 1 LINE WITH OD.OBSTACLE.NAME(OBSTACLE.USED) THUS
            AN *********** OBSTACLE HAS BEEN PLACED ON AN ARC INTO THE NODE
            LET X1 = RN.EASTING(RA.END.1.NODE(POINTER))
            LET Y1 = RN.NORTHING(RA.END.1.NODE(POINTER))
            LET X2 = RN.EASTING(RA.END.2.NODE(POINTER))
            LET Y2 = RN.NORTHING(RA.END.2.NODE(POINTER))
            LET EO.EASTING(OBSTACLE) = X1 + (X2 - X1)*.5
            LET EO.NORTHING(OBSTACLE) = Y1 + (Y2 - Y1)*.5
         ALWAYS
      LOOP
     ''SINCE WE MOVED AN OLD UNIT, PUT PRESENT UNIT BACK IN AVAIL SET
      LET RA.COLOR.ME(BEST.TIME.POINTER) = GR.GREEN
      PRINT 1 LINE WITH MOVED.SUA, THE.NODE THUS
      XXXXX *****.** SUA ADDED TO EVERY ARC AROUND NODE ********* XXXX
      EXIT
   ALWAYS
''THE CASE WHERE ONLY ONE NODE IS FLAGGED
   IF RN.FORCE.ADJACENT.TO.NODE(END.1) GT 0
```

```
OR
RN. FORCE. ADJACENT. TO. NODE(END. 2) GT 0
PRINT 1 LINE THUS
XXXXX ONLY ONE NODE HAS ADJACENT ARC (S) WITH FORCES XXXX
IF RN. FORCE. ADJACENT. TO. NODE(END. 2) GT 0
   LET THE. NODE =  END. 2
ALWAYS
IF RN. FORCE. ADJACENT. TO. NODE(END. 1) GT 0
   LET THE. NODE = END. 1
ALWAYS
FOR EACH TEST. ARC  IN RN. END. 2. NODE. SET(THE. NODE)
   DO
   IF TEST. ARC = BEST. ARC
      CYCLE
   OTHERWISE
   IF RA. FORCE. UNIT. SET(TEST. ARC) IS EMPTY
      CYCLE
   OTHERWISE
   IF FU. SUA(F. RA. FORCE. UNIT. SET(TEST. ARC)) GT THE. SUA
      LET THE. FORCE = F. RA. FORCE. UNIT. SET(TEST. ARC)
      LET THE. SUA = FU. SUA(THE. FORCE)
      LET BEST. ARC = TEST. ARC
   ALWAYS
LOOP
FOR EACH TEST. ARC  IN RN. END. 1. NODE. SET(THE. NODE)
   DO
   IF TEST. ARC = BEST. ARC
      CYCLE
   OTHERWISE
   IF RA. FORCE. UNIT. SET(TEST. ARC) IS EMPTY
      CYCLE
   OTHERWISE
   IF FU. SUA(F. RA. FORCE. UNIT. SET(TEST. ARC)) GT THE. SUA
      LET THE. FORCE = F. RA. FORCE. UNIT. SET(TEST. ARC)
      LET THE. SUA = FU. SUA(THE. FORCE)
      LET BEST. ARC = TEST. ARC
   ALWAYS
LOOP
REMOVE THE. FORCE FROM RA. FORCE. UNIT. SET(BEST. ARC)
IF N. RA. FORCE. UNIT. SET(POINTER) LT 1
   LET RN. FORCE. ADJACENT. TO. NODE(RA. END. 1. NODE(BEST. ARC)) = 0
   LET RN. FORCE. ADJACENT. TO. NODE(RA. END. 2. NODE(BEST. ARC)) = 0
ALWAYS
FILE THE. FORCE IN RN. FORCE. UNIT. SET(THE. NODE)
LET FU. EASTING(THE. FORCE) = RN. EASTING(THE. NODE)
LET FU. NORTHING(THE. FORCE) = RN. NORTHING(THE. NODE)
SUBTRACT THE. SUA FROM RA. TOTAL. SUA(BEST. ARC)
LET MOVED. SUA = . 5*THE. SUA
FOR EACH RECEIVING. ARC  IN RN. END. 1. NODE. SET(THE. NODE)
   DO
   ADD MOVED. SUA TO RA. TOTAL. SUA(RECEIVING. ARC)
   LET POINTER = RECEIVING. ARC
   CALL SELECT. OBSTACLE GIVEN POINTER YIELDING ARCDLY, OBSTACLE. USED
   IF OBSTACLE. USED GT 0
      ADD ARCDLY TO RA. OBSTACLE. DELAY. TIME(POINTER)
      CREATE AN ENGINEER. OBSTACLE CALLED OBSTACLE
      LET EO. OBSTACLE. CLASS(OBSTACLE) = OBSTACLE. USED
      LET EO. RA. POINTER. (OBSTACLE) = POINTER
      FILE OBSTACLE IN US. OBSTACLE. LIST(SECTOR)
      FILE OBSTACLE IN RA. OBSTACLE. SET(POINTER)
```

```
            LET BEST.ARC.OBSTACLE.TYPE = OBSTACLE.USED
            PERFORM ENGINEER.ASSET.UPDATE GIVEN BEST.ARC.OBSTACLE.TYPE
            PRINT 1 LINE WITH OD.OBSTACLE.NAME(OBSTACLE.USED) THUS
            AN *********** OBSTACLE HAS BEEN PLACED ON AN ARC INTO THE NODE
            LET X1 = RN.EASTING(RA.END.1.NODE(POINTER))
            LET Y1 = RN.NORTHING(RA.END.1.NODE(POINTER))
            LET X2 = RN.EASTING(RA.END.2.NODE(POINTER))
            LET Y2 = RN.NORTHING(RA.END.2.NODE(POINTER))
            LET EO.EASTING(OBSTACLE) = X1 + (X2 - X1)*.5
            LET EO.NORTHING(OBSTACLE) = Y1 + (Y2 - Y1)*.5
            ALWAYS
    LOOP
    FOR EACH RECEIVING.ARC IN RN.END.2.NODE.SET(THE.NODE)
       DO
       ADD MOVED.SUA TO RA.TOTAL.SUA(RECEIVING.ARC)
       LET POINTER = RECEIVING.ARC
       CALL SELECT.OBSTACLE GIVEN POINTER YIELDING ARCDLY,OBSTACLE.USED
       IF OBSTACLE.USED GT 0
            ADD ARCDLY TO RA.OBSTACLE.DELAY.TIME(POINTER)
            CREATE AN ENGINEER.OBSTACLE CALLED OBSTACLE
            LET EO.OBSTACLE.CLASS(OBSTACLE) = OBSTACLE.USED
            LET EO.RA.POINTER.(OBSTACLE) = POINTER
            FILE OBSTACLE IN US.OBSTACLE.LIST(SECTOR)
            FILE OBSTACLE IN RA.OBSTACLE.SET(POINTER)
            LET BEST.ARC.OBSTACLE.TYPE = OBSTACLE.USED
            PERFORM ENGINEER.ASSET.UPDATE GIVEN BEST.ARC.OBSTACLE.TYPE
            PRINT 1 LINE WITH OD.OBSTACLE.NAME(OBSTACLE.USED) THUS
            AN *********** OBSTACLE HAS BEEN PLACED ON AN ARC INTO THE NODE
            LET X1 = RN.EASTING(RA.END.1.NODE(POINTER))
            LET Y1 = RN.NORTHING(RA.END.1.NODE(POINTER))
            LET X2 = RN.EASTING(RA.END.2.NODE(POINTER))
            LET Y2 = RN.NORTHING(RA.END.2.NODE(POINTER))
            LET EO.EASTING(OBSTACLE) = X1 + (X2 - X1)*.5
            LET EO.NORTHING(OBSTACLE) = Y1 + (Y2 - Y1)*.5
            ALWAYS
    LOOP
    PRINT 1 LINE WITH MOVED.SUA, THE.NODE THUS
    XXXX *****.** SUA ADDED TO ALL ARCS AROUND NODE ********** XXXXX
  ''SINCE WE MOVED AN OLD UNIT, PUT PRESENT UNIT BACK IN AVAIL SET
    LET RA.COLOR.ME(BEST.TIME.POINTER) = GR.GREEN
    EXIT
ALWAYS
'PLACE.UNIT'
IF UNIT GT 0
      FILE UNIT IN RA.FORCE.UNIT.SET(BEST.TIME.POINTER)
ALWAYS
LET X1 = RN.EASTING(END.1)
LET Y1 = RN.NORTHING(END.1)
LET X2 = RN.EASTING(END.2)
LET Y2 = RN.NORTHING(END.2)
IF UNIT GT 0
    LET FU.EASTING(UNIT) = X1 + (X2 - X1)*.5
    LET FU.NORTHING(UNIT) = Y1 + (Y2 - Y1)*.5
ALWAYS
IF OBSTACLE.USED GT 0
   LET EO.EASTING(OBSTACLE) = X1 + (X2 - X1)*.5
   LET EO.NORTHING(OBSTACLE) = Y1 + (Y2 - Y1)*.5
ALWAYS
IF BEST.TIME.POINTER GT 0
    LET RN.FORCE.ADJACENT.TO.NODE(RA.END.1.NODE(BEST.TIME.POINTER)) = 1
```

```
        LET RN.FORCE.ADJACENT.TO.NODE(RA.END.2.NODE(BEST.TIME.POINTER)) = 1
        LET RA.DURATION(BEST.TIME.POINTER) = BEST.TIME
        ADD BEST.ARC.OBSTACLE.DELAY.TIME TO
                                    RA.OBSTACLE.DELAY.TIME(BEST.TIME.POINTER)
        IF UNIT GT 0
           ADD FU.SUA(UNIT) TO RA.TOTAL.SUA(BEST.TIME.POINTER)
        ALWAYS
    ALWAYS
    IF BEST.ARC.OBSTACLE.TYPE LE 0
       LET BEST.ARC.OBSTACLE.DELAY.TIME = 0
    ELSE
       LET BEST.ARC.OBSTACLE.DELAY.TIME =
                     OD.DELAY.CAUSED(BEST.ARC.OBSTACLE.TYPE)
    ALWAYS
    IF UNIT LE 0
       EXIT
    OTHERWISE
    IF BEST.ARC.OBSTACLE.TYPE NE 0
       PRINT 10 LINES WITH FU.SUA(UNIT),BEST.TIME.POINTER,
          RA.TOTAL.SUA(BEST.TIME.POINTER),
          OD.OBSTACLE.NAME(BEST.ARC.OBSTACLE.TYPE),
          BEST.ARC.OBSTACLE.DELAY TIME,
          RA.OBSTACLE.DELAY.TIME(BEST.TIME.POINTER),
          RA.DURATION(BEST.TIME.POINTER) THUS

             - XXXXXXX RESULTS XXXXXXX

ADDITIONAL  ****.**      SUA ADDED TO ARC          *********
TOTAL SUA ON ARC =                                 ******.**
OBSTACLE TYPE ADDED =                               ********
OBSTACLE DELAY TIME ADDED =                 *****.** MINUTES
TOTAL OBSTACLE DELAY TIME THIS ARC =        *****.** MINUTES
TOTAL BATTLE DELAY TIME THIS ARC =          *****.** MINUTES

    ELSE
       PRINT 10 LINES WITH FU.SUA(UNIT),BEST.TIME.POINTER,
          RA.TOTAL.SUA(BEST.TIME.POINTER),
          RA.OBSTACLE.DELAY.TIME(BEST.TIME.POINTER),
          RA.DURATION(BEST.TIME.POINTER) THUS

                 XXXXXXX RESULTS XXXXXXX

    ADDITIONAL  ****.**      SUA ADDED TO ARC          *********
    TOTAL SUA ON ARC =                                 ******.**
    NO OBSTACLE ADDED
    NO OBSTACLE DELAY TIME ADDED
    TOTAL OBSTACLE DELAY TIME THIS ARC =        *****.** MINUTES
    TOTAL BATTLE DELAY TIME THIS ARC =          *****.** MINUTES

    ALWAYS
    FOR EACH POINTER IN BEST.TIME.LIST
        DO
        REMOVE POINTER FROM BEST.TIME.LIST
    LOOP
    FOR EACH ARC.SRG IN AOA.ROUTE(AVE.OF.APPR)
        DO
        ADD 1 TO AS.COUNTER
        LET AS.TIME(ARC.SRG)  = 0
        LET AS.OBSTACLE.TYPE(ARC.SRG)  = 0
    LOOP
```

```
''  UPDATE ENGINEER ASSET ACCOUNT
    IF BEST.ARC.OBSTACLE.TYPE GT 0
       PERFORM ENGINEER.ASSET.UPDATE GIVEN BEST.ARC.OBSTACLE.TYPE
    ALWAYS
    'OUT'
    RETURN
    END      ''OF ROUTINE DEPLOY.ASSETS.IN.SECTOR
```

LIST OF REFERENCES

1. Craig, Dean, A Model for the Planning of Manuever Unit and Engineer Asset Placement, M.S. Thesis, Naval Postgraduate School, Monterey, California, October, 1985.

2. Boyd, Kenneth D., Methodologies of Direct Fire Allocation and Maneuver Unit Allocation and Placement, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1985.

3. Kazimer, Robert V., Combat Engineer Allocation Model, M.S. Thesis, Naval Postgraduate School, Monterey, California, December 1984.

4. Krupenevich, Thomas P., Network Representation for Combat Models, M.S. Thesis, Naval Postgraduate School, Monterey, California, December 1984.

5. Manzo, Joseph J., Design Considerations for Tactical Planning Networks in the Advanced Land-Air Model (ALARM), Naval Postgraduate School, Monterey, California, July 1985.

6. Taylor, James G., Force-on-Force Attrition Modelling, Military Applications Section, Operations Research Society of America, Arlington, Virginia, 1981

7. Kilmer, Robert, Using the Generalized Value System for Future State Decision Making, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1986.

8. Lindstrom, Robin, Field Artillery Module for the AirLand Research Model, M.S. Thesis, Naval Postgraduate School, Monterey, California, to be published June 1986.

9. Finley, Leonard M., Field Artillery Module for the AirLand Research Model, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1986.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Superintendent Attn: Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Deputy Undersecretary of the Army for Operations Research Room 2E261, Pentagon Washington, D.C. 20310	2
4.	Director Attn: Dr. Wilbur Payne U.S. Army TRADOC Operations Research Agency White Sands Missile Range, New Mexico 88002	1
5.	Commander U.S. Army Combined Arms Center Attn: BG David Maddox Fort Leavenworth, Kansas 66027	1
6.	Director Attn: Mr. E. B. Vandiver III U.S. Army Concepts Analysis Agency Bethesda, Maryland 20814	1
7.	Bell Hall Library U.S. Army Combined Arms Center Fort Leavenworth, Kansas 66027	1
8.	Dr. Samuel H. Parry, Code 55Py Department of Operations Research Naval Postgraduate School Monterey, California 93943	5
9.	Dr. Arthur Schoenstadt, Code 532h Department of Mathematics Naval Postgraduate School Monterey, California 93943	5
10.	Director Attn: COL Anthony Brinkley Studies and Analysis Directorate Headquarters, U.S. Army TRADOC Fort Monroe, Virginia 23651	1
11.	Captain Joseph McLaughlin 2812 South 14th Street Leavenworth, Kansas 66048	2

217506

Thesis
M2423 McLaughlin
c.1 The extension of
unit allocation and
countermobility planning algorithms in the
Airland Research Model.